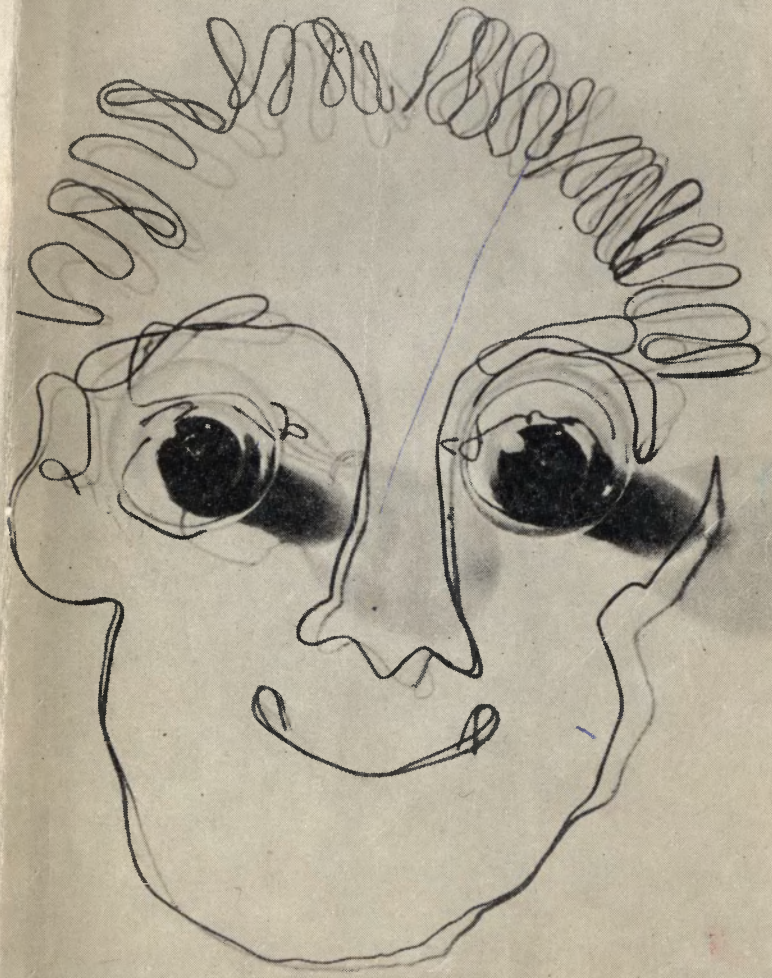
В МИРЕ НАУКИ И ТЕХНИКИ

НТ

С. ГАСС

ПУТЕШЕСТВИЕ В СТРАНУ ЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ




ИЗДАТЕЛЬСТВО
МИР

SAUL I. GASS

AN ILLUSTRATED GUIDE TO LINEAR PROGRAMMING

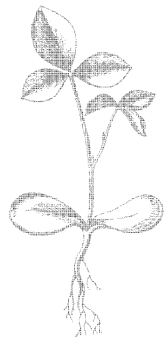
McGraw-Hill Book Company 1970

С. ГАСС

ПУТЕШЕСТВИЕ В СТРАНУ ЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ

Перевод с английского
Ю. Н. Сударева

Предисловие Ю. В. Овсиенко



ИЗДАТЕЛЬСТВО «МИР» МОСКВА 1973

517.8
Г22

Гасс С.

Г22 Путешествие в Страну Линейного Программирования. Пер. с англ. Ю. Н. Сударева. Предисл. Ю. В. Овсиенко. М., «Мир», 1971
176 стр. с илл. (В мире науки и техники)

Почему самые разные специалисты вынуждены прибегать к математическим методам оптимального управления и, в частности, к линейному программированию? Как от сугубо практической задачи перейти к ее математической модели? Как соотносится эта модель с реальной действительностью? Каковы возникающие при этом трудности? На все эти вопросы в доступной и занимательной форме отвечает в настоящей книге крупный американский ученый С. Гасс, уже известный советскому читателю по своей монографии «Линейное программирование».

Книга представляет интерес для самого широкого круга читателей — от школьников старших классов до руководителей предприятий и организаций.

Г $\frac{3314-177}{041(01)-73}$ 517.8

Редакция научно-популярной и научно-фантастической литературы

ПРЕДИСЛОВИЕ

Если вас заинтересует какая-то проблема современной физики, химии, биологии или любой другой из естественных наук, вы без труда найдете научно-популярную книгу, которая не только удовлетворит ваше любопытство, но, возможно, даже подтолкнет к более глубокому изучению этой области. И если, прочитав книгу, скажем, по физике, вы так и не постигнете до конца той же теории относительности, то все равно непременно почувствуете, как интересна эта наука.

Иное дело с наукой об управлении, и прежде всего об управлении различными процессами общественной жизни. Хотя сегодня популярных книг, брошюр и статей по проблемам управления издается немало, среди них почти нет работ, посвященных современным математическим методам управления. Именно поэтому, если мы спросим у неспециалиста, что он думает о применении математических методов в такой науке, как, например, экономическая кибернетика, то, вероятнее всего, услышим в ответ, что это дело в высшей степени сложное и непонятное да к тому же еще и ужасно скучное.

Предлагаемая советскому читателю книга известного американского исследователя Саула И. Гасса непременно рассеет такое представление. На высоком научном уровне и вместе с тем увлекательно и живо рассказывает автор о математических методах изучения общественной жизни и, в частности, о методах линейного программирования и их практическом приложении.

В чем же сущность методов линейного программирования? Попытаемся коротко ответить на этот вопрос с несколько более общих позиций, чем это сделано в «Путешествии в Страну Линейного Программирования».

В самых разных областях нашей жизни и деятельности мы постоянно сталкиваемся по сути дела с одним и тем же классом задач. Нам известны (полностью или

частично) ситуация, в которой мы находимся, и множество возможных альтернативных вариантов нашего дальнейшего поведения. Естественно возникает вопрос: какой из вариантов выбрать, а от каких отказаться? Грубо говоря, в решении этого вопроса и заключается проблема управления. Она с равным основанием может относиться к поведению каждого отдельного человека или группы людей, к развитию тех или иных хозяйственных процессов или экономики в целом, к разработке каких-то операций и т. п. Итак, необходимо остановиться на одном из допустимых вариантов, то есть отыскать план. Здравый смысл подсказывает нам, что нужно избрать наилучший вариант, который принесет наибольшую выгоду. Однако сам по себе вопрос «Какой вариант выбрать?» ни на иоту не приближает нас к решению задачи управления. Более того, он ее даже и не формулирует, поскольку остается совершенно неясным, что означают слова «наилучший вариант», «наибольшая выгода». Определение этого и составляет одну из важнейших проблем, без решения которой невозможны ни постановка, ни решение задачи управления. По существу речь идет о проблеме цели развития управляемой системы. Цели могут быть, вообще говоря, самыми разнообразными. Применительно к хозяйственному объекту в качестве цели могут выступать требования обеспечить максимум выпускаемой продукции при известном объеме имеющихся ресурсов, минимум затрат на производство фиксированного набора продуктов, максимум дохода и т. п. Если цель сформулирована, то тем самым и понятие «наилучший» становится четко определенным. Наилучший или, как обычно говорят, оптимальный вариант отвечает двум требованиям. Во-первых, он является одним из допустимых (или возможных) и, во-вторых, обеспечивает максимум, или минимум (по смыслу) поставленной цели.

Таким образом, общий смысл широкого круга задач управления достаточно прост, хотя сами они порой бывают чрезвычайно сложными и не всегда поддаются решению на основе имеющихся в настоящее время методов. Именно эти соображения легли в основу целого ряда исследований в области так называемого математического программирования.

Постановка задачи управления средствами математического программирования заключается в следующем.

Формально записывается совокупность условий деятельности управляемого объекта, которая называется системой ограничений. Например, для производственного предприятия задаются объемы ресурсов, которые оно может использовать (не обязательно полностью), а также возможности производственных процессов, с помощью которых ресурсы превращаются в продукты. Интенсивности использования производственных процессов — искомые переменные задачи. По физическому смыслу интенсивности производственных процессов — величины неотрицательные. Нулевая величина интенсивности означает, что процесс не происходит.

Следовательно, совокупность условий деятельности управляемого объекта записывается в виде системы уравнений и неравенств. Она определяет множество допустимых вариантов его дальнейшего развития.

Выбор оптимального варианта осуществляется с помощью так называемой целевой функции. Например, мы стремимся получить максимум дохода от реализации произведенной продукции, причем цены на нее известны.

Если сформулирована система ограничений и целевая функция, это значит, что задача управления поставлена. Теперь следует искать ее оптимальное решение, то есть в нашем примере — определить, какие именно производственные процессы надо применять и какова должна быть интенсивность каждого из них, чтобы не допустить перерасхода закрепленных за предприятием ресурсов и в то же время обеспечить максимальную из возможных величину дохода. Решив задачу, мы тем самым найдем оптимальный план развития данного предприятия.

Как же решать такую задачу? Простейший метод, казалось бы, напрашивается сразу. Стоит только, рассмотрев все допустимые варианты, подсчитать, какую прибыль дает каждый из них, и станет ясно, какой вариант дает максимум. Однако на поверку этот метод оказывается самым сложным, а чаще всего просто нереальным, поскольку в большинстве задач математического программирования число допустимых вариантов или практически необозримо, или бесконечно. В то же время получить ответ непосредственно (или, как обычно говорят математики, аналитически), как правило, нельзя. Так что перебрать варианты — идея не столь и абсурдная. Она легла в основу многих алгоритмов, в частности сим-

плекс-метода, о котором пишет С. Гасс. Только перебирать в этих методах нужно не все варианты, а лишь малую их часть.

Таков общий смысл задач математического программирования. Если система ограничений и целевая функция задачи математического программирования линейны, то есть представляют собой уравнения и неравенства первой степени, то мы имеем дело с задачей линейного программирования. Линейное программирование — наиболее глубоко теоретически исследованная область математического программирования, имеющая к тому же огромное прикладное значение, ибо бóльшая часть конкретных задач управления, практически решаемых на ЭВМ, — это задачи линейного программирования.

Линейное программирование зародилось в нашей стране. В работе выдающегося советского математика и экономиста лауреата Ленинской премии академика Л. В. Канторовича «Математические методы организации и планирования производства», изданной еще в 1939 году, впервые была сформулирована задача линейного программирования, описывающая реальную экономическую ситуацию, и разработан алгоритм ее решения. Более того, Л. В. Канторович первым выдвинул идею оптимизации планирования в рамках всего народного хозяйства, породившую целое направление научных исследований. Так что автор «Путешествия» ошибочно относит зарождение линейного программирования к концу 40-х годов. Эта дата характеризует лишь начало интенсивных исследований в области линейного программирования, связанных с исследованием операций в США. Исследование операций ставит своей целью определить, как необходимо организовать работу управляемого объекта с тем, чтобы деятельность его приносила максимальный эффект, и дать соответствующие рекомендации руководителям объекта. Обычно этими вопросами занимается группа экспертов, в состав которой входит математик, формулирующий модель объекта или, если объект достаточно сложен, модели тех или иных его составных частей. В США исследование операций зародилось в годы второй мировой войны прежде всего для решения военных вопросов. Именно военные (а не хозяйственные, как в нашей стране) проблемы и послужили толчком к развитию в США и линейного программирования. Даже такие, казалось бы, невинные задачи,

как составление оптимальных смесей орехов или определение наиболее дешевого рациона для откорма скота, первоначально возникли при планировании военных операций. Мирные их приложения появились гораздо позднее, когда выяснилось, что использование линейного программирования способно дать большой экономический эффект и для решения самых разных хозяйственных вопросов. Группы по исследованию операций превратились в специализированные консультативные фирмы, обслуживающие предприятия. По свидетельству Ф. Бурлацкого, в США «сейчас имеется 150 консультативных фирм, которые берут подряды на разработку конкретных программ для тех или иных фирм, концернов, предприятий. Эти консультативные организации сейчас продают свои услуги клиентуре Западной Европы и других районов мира. Около 20 тысяч ведущих компаний США и сотни крупнейших фирм и концернов в других странах пользуются услугами консультативных фирм» [1].

Круг конкретных задач, решаемых методами линейного программирования, за последние годы необычайно расширился, о чем весьма занимательно сумел рассказать С. Гасс.

Однако автор не осветил ряда важных сторон линейного программирования. Обсуждая методы решения задач, С. Гасс не дает их экономической интерпретации. Между тем процесс нахождения оптимального решения имеет достаточно прозрачный экономический смысл, постигнуть который следует отнюдь не для удовлетворения праздного любопытства. Знание экономического содержания алгоритма помогает лучше понять, как принимает решения плановик, не использующий математические методы и ЭВМ, в какие его рассуждения могут вкрасться ошибки и как их преодолеть даже в тех случаях, когда по тем или иным причинам нельзя построить математическую модель объекта.

В книге лишь вскользь упоминается о так называемой двойственной задаче и ее переменных, в то время как свойства переменных двойственной задачи позволили сделать вывод о том, что эти величины — важный инструмент ценообразования. Исследованию вопросов

---

[1] Ф. Бурлацкий, Надежды и иллюзии, «Новый мир», 1972, № 7, стр. 154.

ценообразования с позиций математического программирования сегодня уделяется особое внимание.

И наконец, в книге нет ни слова о такой важной проблеме, как оптимизация планирования и управления народным хозяйством в масштабе всей страны. Эта проблема, конечно, выходит за рамки линейного программирования, однако многие возникающие вопросы аналогичны тем, с которыми сталкивается исследователь, пытающийся отыскать оптимальный план развития экономического объекта. Причины этого понятны и связаны с характером общественного строя в США.

Таким образом, хотя в книге С. Гасса большинство примеров взято из экономики, ее нельзя назвать экономической. Вместе с тем основы техники моделирования локальных экономических процессов средствами линейного программирования и принятия на этой базе оптимальных решений рассмотрены достаточно серьезно. Следует только помнить, что социально-экономическая природа общественного строя США накладывает определенный отпечаток на характер многих конкретных задач. Поэтому их нельзя механически переносить на наши условия. Однако принцип построения модели объекта, а также тип математической задачи и метод ее решения остаются одними и теми же.

От читателей книги не требуется сколько-нибудь серьезного знания математики. Поэтому их круг может быть весьма обширен. В первую очередь хочется рекомендовать книгу молодежи, ищущей приложения творческим силам. Большую помощь она окажет инженерам, экономистам, плановикам, а также всем, кто хочет получить первое представление об одном из важнейших методов, которым располагает современная теория управления.

Надеюсь, что путешествие по Стране Линейного Программирования будет действительно увлекательным и несомненно поможет «прорубить окно» в сложный и интересный мир одной из областей современной науки.

В заключение хочется привести слова одного из величайших математиков мира — Леонарда Эйлера. Он был убежден в том, что «в мире не происходит ничего, в чем не был бы виден смысл какого-нибудь максимума или минимума». Так что возможности познания мира методами оптимизации поистине неисчерпаемы.

Ю. Овсиенко

Искусство обучения есть искусство будить в юных душах любознательность и затем удовлетворять ее, а здоровая живая любознательность бывает только при хорошем настроении, когда же насильно забивают голову знаниями, они только гнетут и засоряют ум. Чтобы переварить знания, надо поглощать их с аппетитом.

Анатоль Франс

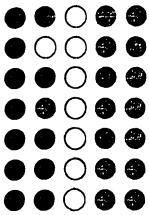
ПРЕДИСЛОВИЕ

...начала вещей недоступны
для глаза...

Наука об управлении и арсенал технических приемов и методов, с ней связанных, привлекают к себе внимание довольно широкой и все увеличивающейся аудитории. Поскольку область линейного программирования представляет собой ключевой элемент мощной и разветвленной науки об управлении, я счел уместным написать элементарную книгу, которая была бы полезна этой аудитории во многих отношениях. Для одних эта книга, я надеюсь, сыграет роль катализатора и послужит переходом к специальной литературе, другие просто ограничатся ею, познакомившись с тем, что их интересовало. Но во всех случаях я хочу надеяться, что час или два, проведенные в «путешествии» (как и некоторые размышления над довольно ограниченным математическим аппаратом, в нем использованным), помогут читателям уловить самые существенные черты линейного программирования. Таким образом, цель «Путешествия в Страну Линейного Программирования» состоит в том, чтобы познакомить читателей на элементарном уровне, но в то же время достаточно серьезно, хотя и не без занимательности, с основными понятиями одной из областей современной науки.

Поскольку я активно занимаюсь линейным программированием с 1952 года, мне трудно сейчас точно выделить из многочисленных факторов именно те, которые повлияли на окончательный вариант данной книги. Ее зародыш можно обнаружить в публикации 1954 года, которая не носила специального характера. Все последующие годы мой интерес к теме поддерживался и развивался благодаря книгам, лекциям, встречам, конференциям и, безусловно, беседам с друзьями и коллегами. Я с благодарностью признаю их огромный вклад. Я хочу также выразить глубокую признательность Биллу Маквильяму за превосходные иллюстрации, которые как нельзя лучше отражают суть книги.

Саул И. Гасс



ВВЕДЕНИЕ

(О том, как надо одеваться, о прямых линиях и о первых шагах по Стране Линейного Программирования)

Поиск наилучшего решения (максимума, минимума или вообще решения, оптимального в том или ином смысле) занимал умы людей на протяжении многих веков. Еще Евклид описал способы построения наибольшего и наименьшего из отрезков, соединяющих данную точку с окружностью, и показал, как среди параллелограммов с заданным периметром найти параллелограмм максимальной площади. Великие математики XVII и XVIII веков развили новые методы оптимизации, которые позволили решить целый комплекс задач из области геометрии, механики и физики. К числу таких задач, например, относится отыскание минимальных поверхностей вращения или кривой наибыстрейшего спуска.

В наше время возник совершенно новый класс задач, связанный с теми сложными организационными структурами, которые столь часто встречаются в современном обществе. (Наша естественная склонность ставить и решать подобные задачи проявляется в выражениях типа «с наименьшими затратами», «полная отдача», «максимальная прибыль» и т. п.) Сюда относятся как вопросы наиболее эффективного управления хозяйством или предприятием, оптимального развития отрасли, так и совсем «земные» задачи, вроде составления самого дешевого и рационального корма для крупного рогатого скота. Попытки точно сформулировать и решить подобные задачи привели к созданию новых важных методов оптимизации. Один из этих методов — линейное программирование — и составляет предмет данной книги. Определить его, не прибегая к специальной терминологии, можно двумя путями: объяснив буквальное значение выражения «линейное программирование» или просто перечислив типичные задачи, которые решаются с его помощью. Поскольку каждый из них обладает своими преимуществами, мы воспользуемся обоими.

В самом широком смысле слова программирование, линейное или иное, имеет дело с задачами о наиболее эффективном использовании, или распределении, ограниченных ресурсов. Такие задачи занимают центральное

место в экономике. Однако они возникают не только в промышленности и хозяйстве, но и в повседневной жизни каждого человека, появляясь в самых разных обличьях.

О ТОМ, КАК НАДО ОДЕВАТЬСЯ

Ай да наш малютка Джон!
Спать в носках собрался он,
И, забыв ботинок снять,
Он улегся на кровать.

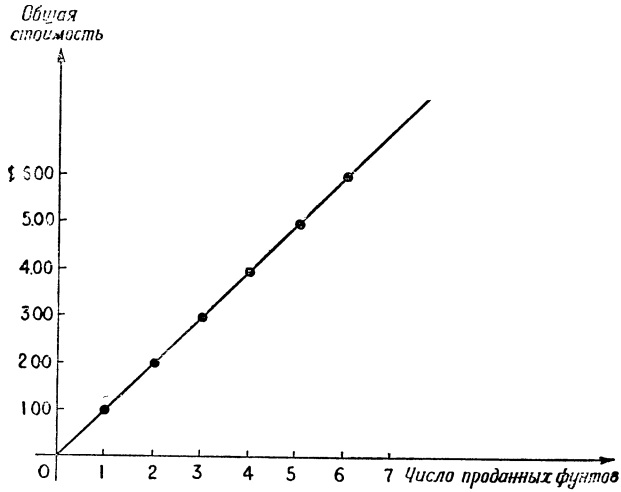
Первое, с чем мы сталкиваемся ежедневно поутру, — это задача программирования утреннего одевания. Мы должны выбрать программу действий, которая позволит

нам одеться, причем так, чтобы выполнялись определенные ограничения, или общепринятые правила (носки надевать необходимо, но не поверх ботинок и т. д.). Время — наш основной ресурс, и выбранная программа должна быть наилучшей в том смысле, в каком каждый понимает свой расход утреннего времени.

Лично я, отбрасывая «несущественные» детали, включаю в свою программу шесть предметов одежды: ботинки, носки, брюки, рубашку, галстук и пиджак. Программа действий представляет собой любой порядок, в котором можно надеть эти предметы. Всего в этом случае существует $6 \times 5 \times 4 \times 3 \times 2 \times 1 = 720$ различных программ. Многие из них недопустимы, так как либо не удовлетворяют общепринятым ограничениям (носки поверх ботинок), либо непрактичны (галстук под рубашкой). Но даже после того, как эти недопустимые

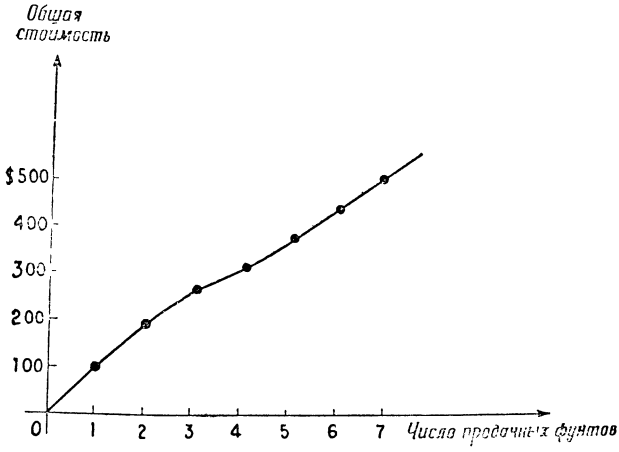
решения будут отброшены, мне все равно придется исследовать некоторое количество допустимых программ.

Как же выбрать окончательное, оптимальное решение? В этой, как и в последующих задачах, имеется некоторая мера эффективности, некая основная цель, позволяющая нам сравнивать эффективность допустимых программ. Если я смогу как-нибудь сравнить между собой меры наших программ, то я сумею тем самым выбрать из них оптимальную. В данной конкретной задаче я хочу минимизировать время, необходимое для того, чтобы одеться. Это моя мера эффективности (на языке



программирования — целевая функция), с помощью которой я могу сравнивать между собой различные допустимые порядки. Честно говоря, я не решил эту задачу с точностью до секунды, однако в течение многих лет мой оптимальный порядок таков: носки, рубашка, брюки, галстук, ботинки, пиджак. Это мое оптимальное решение, оно минимизирует время, необходимое, чтобы одеться, не нарушая общепринятых ограничений. Кто-то другой, возможно, воспользуется иной целевой функцией, например захочет как можно меньшее число раз открыть и закрыть ящики и дверцы в шкафу, то есть минимизировать утренний шум. Соответственно его оптимальное решение может отличаться от моего.

Хотя наша задача и не относится к линейному программированию, но она типична для программирования, поскольку имеет много допустимых решений. Если бы допустимое решение было единственным, не возникало бы никакой проблемы вообще и искать это решение было бы неинтересно. В некоторых задачах мы также можем из всех допустимых решений выбирать по крайней мере одно оптимальное. О том, как найти эти допустимые решения и определить среди них оптимальное, мы поговорим позже.



Поскольку мы будем заниматься задачами исключительно линейного программирования, следует подчеркнуть, что они образуют некоторое специальное подмножество задач общего программирования, обычно называемого математическим программированием. Среди последних они выделяются тем, что математически описываются с помощью линейных (то есть изображаемых прямой линией) соотношений. Например, если один фунт кофе стоит 1 доллар и продавец не делает скидок за количество купленного товара, то общая стоимость покупки прямо пропорциональна ее весу в фунтах. Другими словами, соотношение между ними, как показано на рисунке, помещенном на стр. 16, изображается прямой линией. Напротив, если продавец сбрасывает 10 центов со стоимости второго проданного фунта, 20 центов с третьего и т. д. до пятого фунта и, наконец, по 50 центов

с каждого последующего фунта, то кривая стоимости будет нелинейной, как показано на рисунке, помещенном на стр. 17. Итак, задачи линейного программирования — это те из задач общего (математического) программирования, в которых соотношения (различного рода ограничения и целевая функция) линейны. Хотя это условие кажется слишком жестким, тем не менее оно встречается во многих важных случаях и значительно упрощает поиск решения.

Задачи программирования, особенно линейного, возникают в самых разнообразных ситуациях. Существуют стандартные приложения методов программирования в области сельского хозяйства, нефтехимии, бумажной промышленности, транспорта, распределения продукции и контроля над инвентарем, военного дела, техники, экономики и т. д. (см. библиографию в конце книги). Мы рассмотрим некоторые из этих приложений и на их примере поясним основные понятия линейного программирования. Но сначала мы должны сделать небольшое отступление, чтобы уяснить место линейного программирования в общей схеме современной науки об управлении, или исследовании операций.

ИССЛЕДОВАНИЕ ОПЕРАЦИЙ И МОДЕЛИ

Коль строить что-нибудь желание придет,
То прежде план составив, мы затем
Свой замысел рисуем на бумаге,
И лишь увидев дома очертанья,
Должны подумать о его цене.

Определения понятия «исследование операций» (ИО) так же, как и советы влюбленным, всегда находят спрос и имеются в избытке. Они могут быть пространными, детализированными, такими, как «исследование операций представляет собой приложение научных приемов и методов к задачам управления некоторой системой, позволяющее найти оптимальные решения этих задач», или краткими, вроде «исследование операций — это количественный здравый смысл» или «это исследование управления». Для нас ИО означает научный подход к решению задач управления. Однако больше всего нас будет интересовать не формальное определение ИО, а его методология.

Разработку любого проекта, основанного на ИО, можно приблизительно разбить на шесть стадий, частично перекрывающих друг друга и не имеющих четких границ [1]. В большинстве случаев ими будут:

формулировка задачи;
разработка математической модели изучаемой системы;
отыскание решения с помощью этой модели;
проверка данной модели и решения;
уточнение решения;
применение решения на практике.

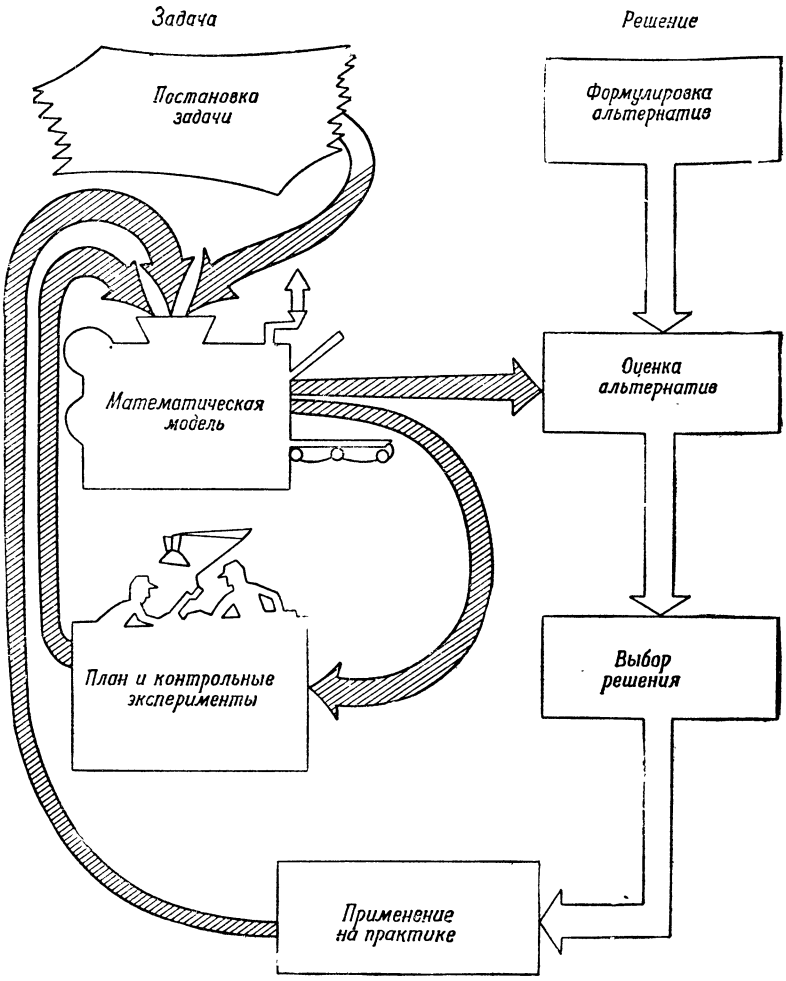
(Здесь мы ввели понятие математической модели, играющей главную роль во всей методологии ИО.) Эти фазы проекта ИО сводятся к осуществлению следующих действий. В каждой задаче мы должны ясно определить цели, поставленные перед системой, изучить обстановку, в которой мы работаем, освоиться с терминологией, людьми и вещами, связанными с нашей задачей, определить различные способы действия, приемлемые для руководителя, дать в устной или какой-то иной форме постановку этой задачи; построить подходящую логическую, или математическую, модель, которая свяжет переменные нашей задачи реалистическими ограничениями и мерой эффективности; найти решение, оптимизирующее эту меру эффективности, то есть допустимое оптимальное решение; сравнить решение, полученное с помощью математической модели, с действительностью, чтобы выяснить, в самом ли деле мы сформулировали и решили именно ту реальную задачу, с которой начали; определить, когда меняется реальная ситуация и какие изменения нужно внести в связи с этим в математическую модель; и, наконец, наиболее важный момент — приложение. Мы должны применить полученное решение на практике (а не только составить отчет) и посмотреть, как оно ведет себя в реальной обстановке. Поскольку наше умение строить математические модели таких задач не достигло еще уровня высокоразвитой науки (многие даже полагают, что это — искусство), то мы должны быть очень чуткими ко всем недостаткам нашего решения и улучшать модель, что приведет к новым решениям, более реалистическим и точным.

---

[1] R. L. Ackoff, The Development of Operations Research as a Science, *J. Oper. Res. Soc. Am.*, 4, № 3 (June 1956).

Существуют три основных типа моделей. *Портретная модель* точно копирует оригинал. Это может быть макет дома, изображение звездного неба в планетарии или идеализированный портрет домашней хозяйки, созданный художником-модельером. *Аналоговая модель* связывает свойства оригинала с другими свойствами, более наглядными и выразительными. Например, понятие температуры описывается графически, причем градус оказывается эквивалентным некоторой специально выбранной единице длины. Наконец, *символическая,* или *логико-математическая, модель* дает символическое описание исследуемой задачи или процесса. Знаменитое уравнение Эйнштейна $e = mc^2$ выражает символически тот факт, что энергия e, содержащаяся в массе m, равна произведению этой массы и квадрата скорости света c^2. Математическая модель означает перевод задачи на язык количественных терминов. Как мы увидим дальше, модель линейного программирования — это математическая модель. В терминах линейного программирования математическая модель представляет собой множество соотношений между переменными, ресурсами, ограничениями и целевой функцией (мерой эффективности). Математическая модель занимает центральное место в методологии ИО. Она позволяет лучше понять исследуемую задачу и процессы, оценить и сравнить между собой взаимоисключающие решения, помогает оценить эффект, который оказывает изменение одной из переменных на все остальные переменные, наконец, она дает нам, хотя и не до конца понятным, а потому несколько таинственным образом, количественную основу, необходимую для того, чтобы мы смогли уточнить и численно выразить наше интуитивное понимание исследуемого процесса.

Следует подчеркнуть, что математическая модель — это первая характерная особенность ИО, или науки об управлении. Математические модели позволяют нам привнести некое подобие научной методологии в те области управления, где до сих пор господствовали интуиция и практический опыт. Насущная потребность в них возникает и при складском хранении оборудования и инвентаря, и при распределении ресурсов, в массовом обслуживании, конфликтных ситуациях, транспорте, промышленных процессах и т. п.


Задача
Решение
Постановка задачи
Формулировка альтернатив
Математическая модель
Оценка альтернатив
План и контрольные эксперименты
Выбор решения
Применение на практике

Роль математической модели в ИО и управлении можно суммарно изобразить на диаграмме, представленной на помещенном здесь рисунке [1]. После того как задача поставлена (то есть выбрана главная мера эффективности), функциональная форма математической модели определена. Поскольку для этого нужно указать, как именно выбранные нами переменные связаны с исходными данными, необходимы некоторые эксперименты, позволяющие выявить правильную структуру. В одних случаях, экспериментируя, мы просто открываем бухгалтерскую книгу и получаем всю необходимую информацию, в других нам потребуется затратить немало сил и средств. В любом из этих случаев между полученными результатами и структурой модели существует обратная связь.

Именно с помощью математической модели задача связана с предлагаемым решением. Теперь наша основная цель состоит в том, чтобы с помощью математической модели и меры эффективности оценить различные решения задачи и выбрать из них то, которое нам нужно. В некоторых задачах это выполняется автоматически с помощью вычислений, использующих нашу математическую модель. Именно так обстоит дело в линейном программировании. В других случаях для этого требуется изобретательность и немалое остроумие. Поскольку практическое применение полученного решения может повлиять на структуру математической модели, нам придется вернуться назад от решения к задаче, и весь процесс повторится сначала. Все это мы проиллюстрируем сейчас на примере одной частной задачи. Мы будем строить ее математическую модель, которая в нашем случае окажется моделью линейного программирования.

ТРАНСПОРТНАЯ ЗАДАЧА

Великое дело — двигаться.

У некого предпринимателя, занимающегося производством и продажей холодильников, есть две фабрики, которые снабжают три его магазина. В начале месяца он

---

[1] H. W. Goode, An Application of a Highspeed Computer to the Definition and Solution of the Vehicular Traffic Problem, *J. Oper. Res. Soc. Am.*, 5, № 6 (December 1957).

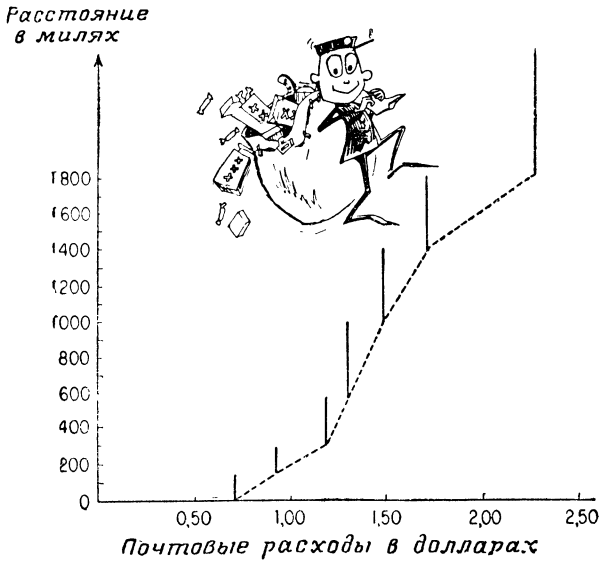
получает от директора каждого из магазинов заявку на определенное число холодильников, в которых данный магазин нуждается в текущем месяце. Совокупность всех таких заявок позволяет установить общее число новых холодильников, которые нужно изготовить на этих двух фабриках. Для простоты мы предположим, что у предпринимателя достаточно ресурсов (рабочей силы, сырья и т. д.), чтобы выполнить все заявки, и что в данный момент отсутствует задел для продажи. Сам по себе процесс производства можно трактовать с помощью математической модели, однако сейчас нас интересует совсем другое. Пусть первому магазину, обозначим его S_1, требуется 10 холодильников, магазину S_2 — 8 и магазину S_3 — 7, всего, таким образом, 25 холодильников. Пусть предприниматель решил изготовить 11 холодильников на первой фабрике F_1 и оставшиеся 14 на фабрике F_2. Задача состоит в том, чтобы выяснить, сколько холодильников нужно отправить с каждой фабрики в каждый магазин, чтобы общая стоимость всех перевозок была минимальной. Основная форма этой задачи, известной под названием *транспортной задачи*, представляет собой одну из самых ранних и наиболее широко используемых формулировок линейного программирования.

Нам нужна дополнительная информация относительно транспортных ограничений и цен. Предположим, что можно отправить любое число холодильников с каждой фабрики в любой магазин, то есть что имеются транспортные пути (железные дороги, шоссе и т. п.), связывающие каждую фабрику со всеми магазинами. Предположим далее, что нам известна также стоимость перевозки одного холодильника с фабрики в магазин в каждом случае. Здесь мы должны принять допущение относительно линейности, которое в некоторых случаях весьма спорно и может вызвать критику. Предположение о линейности, или пропорциональности, состоит в том, что если стоимость перевозки одного холодильника из F_1 в S_1 составляет 10 долларов, то стоимость перевозки двух холодильников составляет 20 долларов, трех — 30 долларов и т. д. Против подобного допущения можно было бы возразить, основываясь на реальном опыте. Если нанять грузовик для перевозки одного холодильника стоит 100 долларов, то в случае двух холодильников стоимость перевозки каждого составит 50 долларов (не считая стоимости погрузочно-разгрузочных работ), в случае

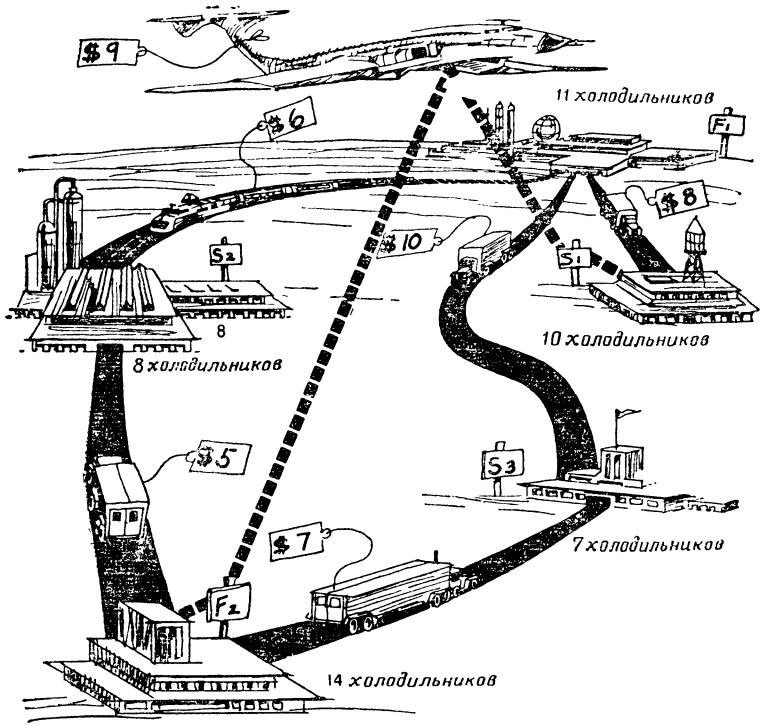
трех — $33^1/_3$ доллара, то есть стоимость меняется нелинейно. В качестве другого примера мы возьмем посылку с объявленной ценностью 3 доллара 80 центов и выясним, во сколько обойдется ее отправка из города Атлантик-Сити. Получится следующая таблица:

Зона	Расстояние от Атлантик-Сити в милях	Стоимость отправки в долларах
Местная	—	0,69
1	50	0,93
2	150	0,93
3	300	1,20
4	600	1,32
5	1000	1,50
6	1400	1,71
7	1800	1,93
8	Свыше 1800	2,25

График зависимости стоимости от расстояния показывает, что почтовые расходы меняются нелинейно. Мы



видим, что график разрывен и составлен из вертикальных, не связанных между собой отрезков. Если бы стоимость изменялась с расстоянием непрерывно, то график совпал бы с пунктирной линией. В большинстве



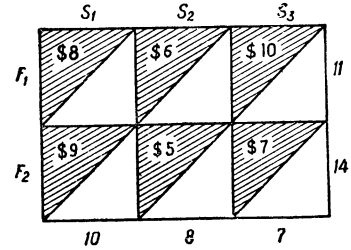
транспортных ситуаций стоимость нелинейна, но с помощью того или иного усреднения можно получить хорошее линейное приближение.

Итак, в нашей задаче мы предполагаем, что стоимость перевозки холодильника между любой фабрикой и любым магазином известна и линейна. На нашем рисунке эта стоимость указана на ярлычках. Например, стоимость перевозки одного холодильника из F_2 в S_3 равна 7 долларам. На рисунке изображены пути, по которым холодильники можно доставить с фабрик в магазины, и представлены все данные, необходимые для решения задачи. В некотором смысле это портретная модель. Хотя такой рисунок и не решает задачи, тем не менее он поможет нам построить подходящую математическую модель. Транспортные задачи имеют, вообще говоря, много допустимых решений. Мы найдем неко-

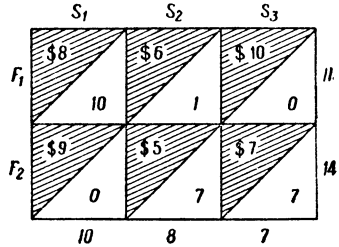
торые из них и одновременно увидим, как линейное программирование применяется к нашей задаче.

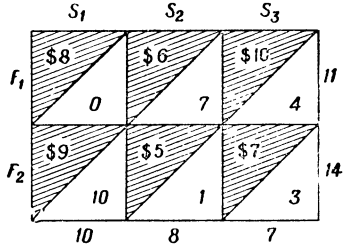
Мы ищем решение, которое удовлетворяет принятым ограничениям (11 холодильников должны быть отправлены из F_1, 14 холодильников — из F_2; S_1 получает 10, S_2 — 8 и S_3 — 7 холодильников) и одновременно минимизирует меру эффективности — общую стоимость всех перевозок.

Чтобы дать математическую постановку задачи, мы сведем всю информацию в таблицу:



Белые треугольники соответствуют неизвестному числу холодильников, которые нужно перевезти с данной фабрики в определенный магазин. Чтобы показать, как легко любой опытный (или даже неопытный) служащий отдела доставки готовой продукции находит решение, мы приведем два возможных варианта:





Читатель без труда сможет найти и другие решения. Числа, стоящие в белых треугольниках таблицы, помещенной слева, соответствуют решению задачи, при котором из F_1 в магазины отправлено $10 + 1 + 0 = 11$ холодильников, а из F_2 $0 + 7 + 7 = 14$ холодильников. Общее число холодильников, полученных магазином S_1 с обеих фабрик, равно $10 + 0 = 10$, магазином S_2 $1 + 7 = 8$ и S_3 $0 + 7 = 7$ холодильников. Второй вариант

решения аналогичен. В предположении, что стоимость перевозок линейна, мы получим, что общая стоимость в первом случае равна

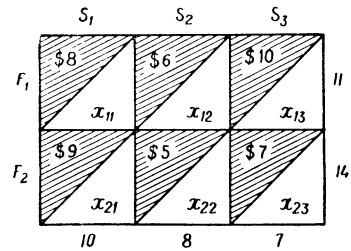
$$8 \times 10 + 6 \times 1 + 5 \times 7 + 7 \times 7 = 170 \text{ (долларов)},$$

а во втором случае

$$6 \times 7 + 10 \times 4 + 9 \times 10 + 5 \times 1 + 7 \times 3 = 198 \text{ (долларов)}.$$

Таким образом, в первом случае затраты на перевозку холодильников меньше, чем во втором, однако не ясно, нет ли другого допустимого решения, позволяющего еще больше снизить затраты на перевозку. Служащему, не пользующемуся методами линейного программирования, не остается ничего другого, как положиться на свой опыт и интуицию. Не перебирая все допустимые решения (как не перебирает их и линейное программирование), служащий просто останавливает свой выбор на некотором частном решении. Разумеется, никакой гарантии, что он нашел абсолютный минимум, у него нет. Методы же линейного программирования позволяют утверждать, что этот минимум существует и его можно найти. В нашей задаче минимум затрат на перевозку холодильников достигается в первом решении.

Для построения математической модели транспортной задачи необходимо ввести некоторые математические сокращения. Пусть x_{11} означает число (пока еще не известное) холодильников, отправленных из F_1 в S_1, x_{12} — число холодильников, отправленных из F_1 в S_2, и т. д. В общем случае x_{ij} — это число холодильников, отправленных с фабрики F_i в магазин S_j. Введем эти обозначения в нашу таблицу:



Теперь очень просто построить математическую модель.

Из F_1 отправлено x_{11}, x_{12} и x_{13} холодильников, всего 11 штук. Аналогично из F_2 отправлено x_{21}, x_{22} и x_{23} холодильников, всего 14 штук. Поскольку требуется всего 25 холодильников $(10+8+7)$ и произведено их ровно 25 $(11+14)$, то все холодильники с каждой фабрики должны быть отправлены в магазины. Таким образом, общее число холодильников, отправленных из F_1, дается уравнением

$$x_{11}+x_{12}+x_{13}=11,$$

а из F_2 —

$$x_{21}+x_{22}+x_{23}=14.$$

Эти суммы получаются путем сложения по строкам чисел, стоящих в таблице.

Поскольку каждый магазин получает столько холодильников, сколько заказывает, то общее количество холодильников, полученное каждым магазином (оно находится суммированием по столбцам), задается уравнениями

$$x_{11}+x_{21}=10 \qquad \text{для } S_1,$$

$$x_{12}+x_{22}=8 \qquad \text{для } S_2$$

и

$$x_{13}+x_{23}=7 \qquad \text{для } S_3.$$

Для каждого набора чисел x_{ij}, где i снова означает номер фабрики, а j — номер магазина, общая стоимость перевозок равна

$$8x_{11}+6x_{12}+10x_{13}+9x_{21}+5x_{22}+7x_{23} \text{ (долларов)}.$$

Эти уравнения представляют собой основные ограничения нашей математической модели. Единственное наше упущение состоит в том, что мы не ограничили x_{ij} так, чтобы они могли принимать только положительные и нулевые значения. Отрицательные x_{ij} означали бы, что холодильники перевозятся из магазина на фабрику, то есть общее число холодильников больше того, которое произведено на фабриках. Мы исключаем такую возможность, вводя ограничения $x_{11} \geqslant 0$ (x_{11} больше или равно нулю), $x_{12} \geqslant 0, \ldots, x_{23} \geqslant 0$ или в общих обозначениях $x_{ij} \geqslant 0$. Эти условия называются в линейном программировании «условиями неотрицательности». Поскольку мы хотим определить множество значений $x_{ij} \geqslant 0$, удовлетворяющих уравнениям и условиям неотрицательности и минимизирующих общую стоимость, то имеем

следующую математическую модель — модель линейного программирования данной транспортной задачи:

найти множество чисел $x_{ij} \geqslant 0$, которые минимизируют выражение

$$8x_{11} + 6x_{12} + 10x_{13} + 9x_{21} + 5x_{22} + 7x_{23} \text{ (долларов)}$$

и удовлетворяют ограничениям

$$\begin{aligned}
x_{11} + x_{12} + x_{13} &= 11,\\
x_{21} + x_{22} + x_{23} &= 14,\\
x_{11} + x_{21} &= 10,\\
x_{12} + x_{22} &= 8,\\
x_{13} + x_{23} &= 7.
\end{aligned}$$

Первое решение, приведенное выше, удовлетворяет этим уравнениям, где $x_{11} = 10$, $x_{12} = 1$, $x_{13} = 0$, $x_{21} = 0$, $x_{22} = 7$ и $x_{23} = 7$, и, как указывалось ранее, это решение дает минимум целевой функции, равный 170 долларам.

Каждая фабрика и магазин порождают уравнение, связывающее между собой те переменные, которые относятся к данной фабрике или магазину. Эти уравнения, так же как и целевая функция, линейны, поскольку они представляют собой просто суммы переменных. Общее количество переменных равно произведению числа фабрик на число магазинов, в нашем случае $2 \times 3 = 6$. Аналогично число уравнений равно сумме числа фабрик и магазинов; в нашем случае оно равно 5. Транспортные задачи могут быть очень громоздкими, однако с вычислительными процедурами (алгоритмами), необходимыми для решения достаточно объемистых задач, справится большинство электронно-вычислительных машин.

С математической точки зрения написанная выше система уравнений довольно любопытна. Во-первых, она содержит одно лишнее уравнение, которое является следствием остальных. Например, если мы отбросим первое уравнение, то его можно получить вновь, вычитая второе уравнение из суммы трех оставшихся уравнений. Важнее, однако, то, что если мы решим нашу задачу, используя стандартные вычислительные методы линейного программирования, переменные x_{ij} в оптимальном решении окажутся целыми числами. До сих пор молча-

ливо предполагалось, что x_{ij} должны быть целыми (мы не в состоянии перевезти $3^3/_4$ холодильника!). Можно строго показать, что если количество фабрик и магазинов выражается целыми числами, то и сама транспортная задача будет допускать оптимальное решение в целых числах. Разумеется, утверждать, что так будет всегда, в любой задаче линейного программирования, нельзя. Возможность решения транспортной задачи в целых числах обусловлена особой структурой уравнений соответствующей ей математической модели.

Нашу модель мы описывали на конкретном примере и поэтому вынуждены были говорить о фабриках, магазинах и холодильниках, однако эти детали не имеют значения. Все рассуждения остаются в силе и в более общем случае, если рассматривать пункты отправления (фабрики), пункты назначения (магазины) и однородные единицы, подлежащие перевозке (холодильники), а также некую меру, которую необходимо минимизировать (общая стоимость перевозок). Мы сейчас проиллюстрируем эту мысль и приведем другой пример линейной программы, называемой транспортной моделью.

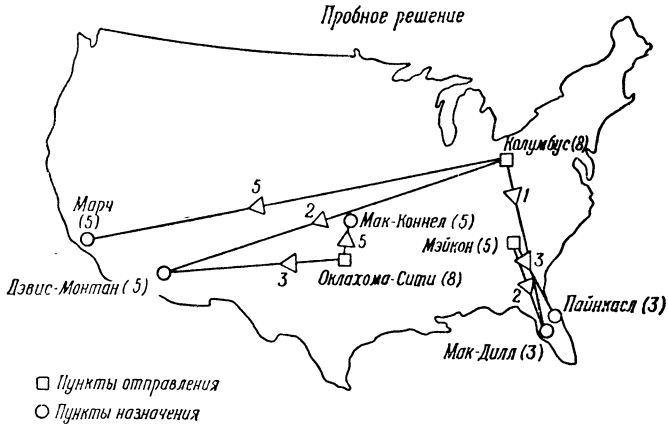
В новой транспортной задаче речь идет о перевозках некоторого груза со складов в определенные пункты назначения. Количество груза на каждом складе ограничено. В пункты назначения необходимое количество груза можно доставить многими способами. Задача состоит в том, чтобы определить такой способ, который не только удовлетворял бы все запросы, но и минимизировал некоторую меру стоимости. Например, мы можем минимизировать одну из следующих целевых функций: общую стоимость в долларах, общую длину всех путей, по которым доставляются грузы, или общее количество транзитных перевозок.

Предположим, что на авиационной базе Лекбурн в Колумбусе прошли испытания большой партии оборудования для самолетов. Вес одного комплекта этого оборудования составляет одну тонну. Теперь необходимо произвести испытания этого оборудования на других базах. Базам Марч, Дэвис-Монтан и Мак-Коннел требуется по пять комплектов этого оборудования, базам Пайнкасл и Мак-Дилл — по три. Из необходимых 21 комплекта 8 имеются в Колумбусе, 8 — на складе в Оклахоме и 5 — в Мэйконе. Требуемое оборудование доставляется в пункты назначения по воздуху. Все необходимые сведения,

включая расстояния между пунктами отправления и назначения, представлены в следующей таблице:

Пункты отправления	Количество имеющихся комплектов	Пункты назначения				
		Мак-Дилл	Марч	Дэвис-Монтан	Мак-Коннел	Пайнкасл
		Требуемое количество комплектов				
		3	5	5	5	3
		Расстояние в милях				
Оклахома-Сити	8	938	1030	824	136	995
Мэйкон	5	346	1818	1416	806	296
Колумбус	8	905	1795	1590	716	854

Требуется минимизировать количество тонно-миль. Оптимальное в этом смысле решение приведено на стр. 33.

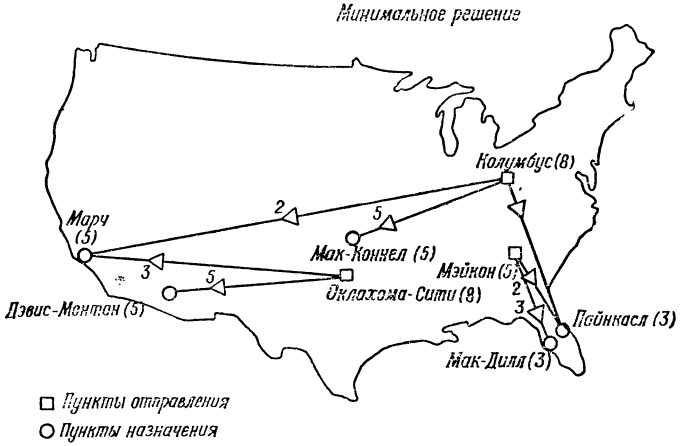


Читателю полезно попытаться самому отыскать наилучший способ перевозок.

Это можно сделать, например, следующим образом. Постараемся удовлетворить запросы баз из ближайших источников. Пусть Мак-Коннел получит свои 5, а Дэвис-Монтан 3 из 5 требующихся комплектов со склада в Оклахома-Сити. Потребности Пайнкасла и Мак-Дилла, расположенных во Флориде, следует по возможности

удовлетворить с базы в Мэйконе. Пусть далее Пайнкасл получит из Мэйкона свои 3, а Мак-Дилл — 2 из 3 требуемых комплектов. Остальные комплекты оборудования следует доставить из Колумбуса. В этой пробной схеме общее число тонно-миль равно 17 792 (см. стр. 32).

Существует много других возможных решений. Минимальное решение (см. рисунок), полученное с помощью методов линейного программирования, дает значение 16 864 тонно-мили. Методы линейного программирования позволяют, исходя из произвольного (например, полученного выше) решения, строить все лучшие и лучшие ре-



шения до тех пор, пока не будет найдено оптимальное, и вместе с тем гарантируют, что минимум будет найден.

Если бы читатель попытался удовлетворить часть потребностей базы Мак-Коннел, доставляя оборудование из Оклахома-Сити (то есть воспользоваться, как это было сделано в первом пробном решении, ближайшим из мест, где имеется необходимое оборудование), то общее количество тонно-миль оказалось бы большим. Этим еще раз подтверждается, что подход к решению, который на первый взгляд кажется наиболее естественным, не обязательно будет самым лучшим.

Рассмотрение транспортной задачи и ее всевозможных разновидностей могло бы послужить темой целого трактата и является важным разделом науки об управлении. Впоследствии мы еще неоднократно встретимся с подобными задачами. Здесь же достаточно сказать,

что математические модели транспортной задачи могут быть крайне разнообразны, в особенности если учесть изобретательность и богатое воображение нового поколения математиков. Однако мне хотелось бы, чтобы читатель обратил внимание на то обстоятельство, что и простой логический подход к постановке задачи может весьма далеко продвинуть нас по заветному пути, ведущему к построению правильной математической модели. В области линейного программирования эта мысль находит особенно яркие подтверждения. Сложные и временами страшно запутанные вопросы здесь обычно удается разрешить, не прибегая к черной магии.

Во время нашего путешествия очень важно порой остановиться, чтобы взглянуть на пройденный путь в перспективе, в его связи со смежными областями. Конец главы — наиболее подходящее для этого место, и автор считает здесь необходимым заявить следующее.

Мне следует умерить свой энтузиазм по отношению к линейному программированию (разделяемый не только мной) при воспоминании о тех трудностях, которые встречаются при формулировке многих операционных задач. Эти трудности возникают тогда, когда мы выбираем нужные ограничения, примиряем противоречивые тенденции, определяем, какие именно данные нужны, и находим их, а также при воздействии психологических и политических моментов, присущих любой ситуации, в которой создается впечатление, что математика и математики якобы собираются заменить тех, кто «на самом деле» разбирается в данной задаче.

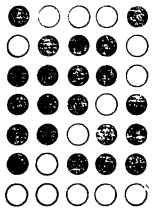
Линейное программирование и его обобщения, будучи сами по себе важной ветвью прикладной математики, являются тем не менее составной частью более общей области науки об управлении — исследования операций. В прошлом создавалось впечатление, что главная задача ИО состоит в оптимизации больших объединенных систем, вроде комплексной системы производства — распределения. Однако в действительности задачи, поддающиеся решению, представляют собой подсистемы подсистем, наподобие упрощенной транспортной задачи. Причины этого обстоятельства многообразны и все еще сохраняют свое влияние — мы не можем построить точную математическую модель комплексных взаимодей-

ствий и не располагаем вычислительной техникой, которая позволила бы нам решать подобные задачи. Таким образом, упрощенное обсуждение задач линейного программирования, которое проводится в этой книге, страдает тем же недостатком, присущим подсистемам, что и ИО в общем случае. Мы можем справиться с важными подклассами задач, однако задачи, связанные с оптимизацией больших систем, все еще ждут своего решения.

Каковы же перспективы? Чувствуется, что ключом к успешным операциям, если не ко всей оптимизации комплексных систем, является развитие информационных систем, использующих ЭВМ и отражающих текущее состояние данной операционной системы. Такая информационная система должна объединять в себе подходящие математические модели и вычислительные методы (вроде линейного программирования) с эвристическими методами, использующими ЭВМ и позволяющими человеку, вооруженному ЭВМ, исследовать те стороны задач, которые не поддаются точному математическому описанию. Например, в задаче о производстве — распределении (подобной той, с которой столкнулся предприниматель, занимающийся производством и сбытом холодильников) мы, быть может, не смогли бы построить математическую модель, показывающую, как нужно распределять ресурсы и оборудование и в каком порядке следует производить единицы продукции. Однако, если продукция уже произведена, мы можем отправить ее в пункты назначения оптимальным способом, используя транспортную модель. В данном случае информационная система показывала бы состояние производства и количество продукции, которую необходимо произвести. Используя эвристические методы, предприниматель мог бы смоделировать различные варианты производства при некоторых условиях и допущениях; определить стоимость каждого варианта; используя транспортную модель, определить для каждого варианта минимальную стоимость перевозок; наконец, выбрать способ, дающий минимальную общую стоимость. Но эта процедура не позволяет нам найти истинный минимум стоимости, поскольку мы предположили, что не обладаем оптимизационной моделью процесса производства. Правда, при высоком быстродействии ЭВМ данный метод позволит предпринимателю испробовать большое количество возможных способов производства (включая и тот, который

обычно используется на его фабриках), прежде чем процесс производства начнется в действительности. Таким образом, он сможет оптимизировать весь процесс в пределах, допускаемых данным методом.

Мы видим, что при таком гипотетическом объединенном информационно-оптимизационно-эвристическом подходе элемент оптимизации представляет собой лишь одну из характерных черт общих методов контроля в комплексных задачах. По-видимому, линейное программирование будет основным строительным блоком при создании подобных методов точно так же, как оно представляет собой основной метод оптимизации во многих более мелких реальных задачах.



ФОРМУЛИРОВКА ЗАДАЧ

(Мы продолжаем наше путешествие и узнаем, как нужно составлять меню, посещаем ферму и неожиданно встречаемся со странными существами, обитающими в Стране Линейного Программирования.)

Основная цель, которую мы преследуем, разрабатывая модель линейного программирования для некоторой операционной задачи, состоит в том, чтобы по начальным данным предсказывать оптимальное решение нашей задачи. При этом предполагается, что мы можем в реальной ситуации (то есть в той настоящей практической задаче, которую хотим решить) правильно определить переменные и ограничения и действовать с ними должным образом. Во многих случаях (практически в большинстве случаев) наша способность точно описывать *подлинную, настоящую задачу* весьма спорна. Однако, как и во всех других областях человеческой деятельности, мы обнаруживаем, что компромиссы, сообразительность и даже некоторая свобода, обусловленная неточностью наших знаний, помогают лучше понять исследуемый процесс и, как мы надеемся, позволяют построить модель, приводящую к улучшенному решению, которое можно применять на практике. Модель, претендующая на точное описание производственных возможностей какой-то фабрики, способна это делать лишь в определенных пределах. Наши предсказания о будущей продукции, которые используются при планировании перевозок, продажи и связанных с этим потребностей, основываются на предполагаемых или измеренных темпах производства, наличии рабочей силы, ресурсов и т. д. Но всякого рода случайности, которые происходят в действительности (здесь — задержка, там — нехватка), влияют на истинное количество продукции. Если модель представляет собой «разумное» математическое описание реальной задачи, то планы, основанные на предсказанной продукции, не уведут нас слишком далеко в сторону и практически позволят нам действовать более эффективно и, следовательно, более прибыльно.

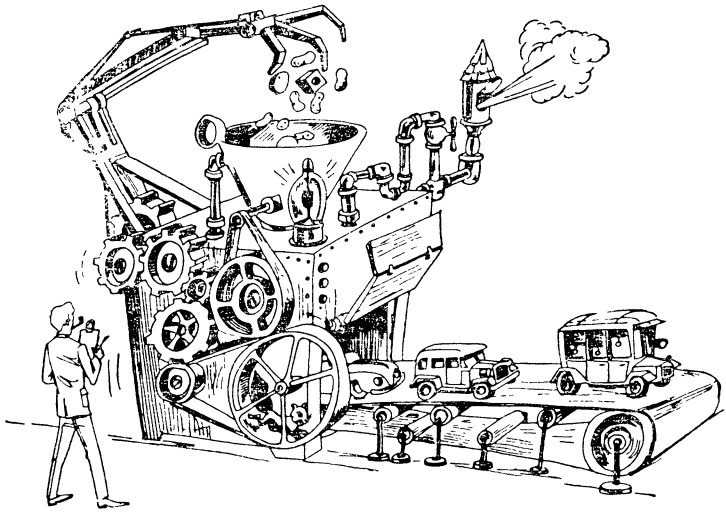
При формулировке задач в терминах линейного программирования нам надо застраховать себя от обвинений в том, что якобы мы предпочитаем инструмент самой работе и, если инструмент не подходит, изменяем саму работу, лишь бы сделать ее пригодной для нашего инструмента. Эта возможная *западня*, однако, не мешает и не должна мешать нам делать подходящие упрощения, когда мы хотим построить модель, верно схватывающую само существо задачи. Наши математические модели должны давать ответы, понятные тем, кто несет ответственность за исследуемые процессы. Ведь этим людям, которых не очень точно называют «управляющими», придется использовать полученные результаты на практике. Эти результаты должны улучшить наши действия (по отношению к подходяще выбранной мере эффективности).

Как мы приходим к формулировке задачи линейного программирования? Какие здесь есть опасные места? Как можно получить нужные нам результаты? Ответы на эти и подобные вопросы нельзя дать в четких и ясных терминах. Самое большее, что я могу здесь сделать, это поговорить о них, чтобы читателю стал ясен их смысл.

Как уже отмечалось, хотя транспортная задача вначале и была определена как задача о перевозке товаров с фабрик в магазины, общая транспортная задача может включать в себя пункты отправления и пункты назначения общего вида. Это обстоятельство следует подчеркнуть особо. На ряде примеров будет показано, какой вид принимает эта задача в разных ситуациях. Читателю (который уже должен осознать, какое тяжкое бремя автор взвалил на его плечи) не следует быть столь ограниченным, чтобы думать, будто окончательная математическая модель не используется нигде, кроме той конкретной ситуации, в которой она возникла. Следующая история поясняет эту мысль.

Еще на заре линейного программирования, где-то около 1953 года, в одной из публикаций существо формулировки задачи объяснялось на примере составления оптимальной смеси продуктов. Владелец фабрики по переработке орехов хочет смешать три сорта орехов. Каждая порция состоит из кэшью, фундука и кедровых орехов, причем должны выполняться некоторые условия. Например, одна порция должна содержать не менее 50% кэшью и не более 25% кедровых орехов. Учитывая воз-

можности фабрики, предприниматель хочет так скомбинировать свои наличные ресурсы орехов, чтобы получить максимальную прибыль. В статье давалось детальное объяснение задачи, математическая модель и числовой пример. Этот пример и статья нашли широкую читательскую аудиторию. И вот когда одного из первых консультантов по вопросам управления пригласил к себе директор крупной автомобильной компании в Детройте, чтобы узнать, где можно познакомиться с новым методом, называемым линейным программированием, консультант порекомендовал именно статью с задачей об орехах. Но



несколько недель спустя, когда консультант заглянул к директору, чтобы узнать, как продвигается курс самообразования, тот выговорил ему за впустую потраченное время. Оказалось, что директор, прочитав и перечитав все, что касалось задачи об орехах, почувствовал себя крупным специалистом по восточным сладостям, но ведь ему нужно было выпускать... автомобили. Директор не смог перевести понятия, относящиеся к задаче об орехах, на язык своего производства. Он не смог «увязать» одно с другим. Я убежден, что современный читатель более проницателен. По крайней мере он предупрежден.

Чтобы показать, насколько гибка модель линейного программирования и как легко она может приспосабливаться к различным ситуациям, я приведу в дальнейшем

целый ряд задач, ставших ныне классическими и описанных в литературе по линейному программированию. Математически все эти задачи аналогичны друг другу и могут быть решены с помощью стандартного вычислительного метода линейного программирования — *симплекс-метода*. Чтобы обосновать этот метод, нужно привлечь довольно сложные математические понятия, знакомство с которыми не входит в задачу настоящей книги[1]. Моя цель состоит в том, чтобы познакомить читателя с наиболее существенными чертами линейного программирования, а не сделать из него специалиста в этой области. Таким образом, вычислительная сторона вопроса будет по своей природе рудиментарной, а ее обсуждение отложено до следующей главы и дополнения.

ЗАДАЧА О ДИЕТЕ

Чтобы узнать, каков пудинг, его надо съесть.

Всякий, кто хочет жить, не выходя за рамки своего бюджета, может многими способами распределить свои средства. Расчетливая хозяйка какую-то сумму откладывает на квартирную плату, а какую-то — на одежду, пищу, развлечения, транспорт и т. п. Легко сосчитать, сколько придется отложить на нужды, подобные квартирной плате, поскольку размер ее постоянен. А вот распределяя деньги на еду и на развлечения, мы обычно основываемся на прошлом опыте и на случайных текущих обстоятельствах. Конкретное распределение средств зависит, однако, и от меры эффективности, которую выбирает хозяйка и в которой учитываются все затраты. В подобных задачах бывает довольно трудно не только оптимизировать, но даже и определить общую меру эффективности. Поэтому обычно пытаются разбить данную задачу на более мелкие и более доступные оптимальные задачи, в каждой из которых своя мера эффективности и связанные с ней ограничения. Рассмотрим, например, такую мелкую задачу, с которой сталкивается хозяйка, собирающаяся накормить свою семью. Мы очень

[1] Читателя, который хочет познакомиться с математической стороной вопроса, мы отсылаем к приложению в конце книги и к учебнику автора: С. Гасс, Линейное программирование (методы и приложения), М., Физматгиз, 1961.

упростим себе задачу (что и будем делать впредь в подобных случаях), если ограничимся случаем, когда наша хозяйка готовит завтрак своим детям.

Забыв о бюджете, озабоченная хозяйка хочет сделать завтрак достаточно питательным. Заглянув в справочник, содержащий сведения о витаминах и калориях, она приходит к выводу, что дети должны получить за завтраком по крайней мере 1 мг тиамина, 5 мг ниацина и 400 калорий. Для удовлетворения этой потребности она располагает двумя видами крупяных изделий: *K* и *C*. Можно выбрать либо один вид, либо другой, либо, наконец, смесь обоих видов. Красивые этикетки на каждой из пачек сообщают среди прочего, что 1 унция *K* содержит 0,10 мг тиамина, 1 мг ниацина и 110 калорий; в то время как 1 унция *C* содержит 0,25 мг тиамина, 0,25 мг ниацина и 120 калорий. Мы видим, что в этой задаче очень легко найти решения, то есть составить различные меню. Нужное количество питательных веществ мы получим, съев 10 унций *K* или 20 унций *C*. А как обстоит дело со стоимостью такой диеты, с бюджетом? Наша хозяйка должна так составить меню, чтобы дети получили необходимое количество питательных веществ при минимальных затратах. Если унция *K* стоит 3,8 цента, а унция *C* — 4,2 цента, то стоимости двух предложенных решений равны соответственно 38 и 84 центам. Первое решение (съесть 10 унций *K*) даст 1 мг тиамина, 10 мг ниацина и 110 калорий, тогда как второе (съесть 20 унций *C*) даст 5 мг тиамина, 5 мг ниацина и 2400 калорий. Первое решение дает нужное количество тиамина, но в нем слишком много других составных частей, а во втором решении ниацина как раз столько, сколько нужно, но в нем много всего остального. Если детям на завтрак захочется только *K*, то хозяйка не сможет взять *K* в количестве, меньшем 10 унций, так как в нем не будет даже 1 мг тиамина. Точно так же, если им захочется только *C*, то они должны будут съесть его 20 унций, поскольку в противном случае не получат достаточного количества ниацина. Единственно возможный вариант, отличный от предыдущих, состоит в том, чтобы взять смесь обоих видов крупяных изделий (хозяйка должна определить, существует ли комбинация, содержащая достаточное количество питательных веществ и более дешевая, чем решение с 10 унциями *K*). Практически ей хочется найти самую

дешевую смесь, удовлетворяющую «ограничениям питательности». Как же она сумеет определить диету минимальной стоимости? И тут на помощь приходит линейное программирование!

В приведенных выше рассуждениях, говоря о смешивании двух видов изделий, мы молчаливо допускали

некоторые линейные соотношения. Во-первых, мы считали, что количество принятых питательных веществ пропорционально количеству съеденной пищи. Так, например, 1 унция K содержит 110 калорий, в то время как 2 унции содержат 220 калорий и т. д. Точно так же количество питательных веществ, содержащихся в одном виде пищи, складывается с аналогичным количеством, содержащемся в другом виде. Например, в 1 унции K и 1 унции C всего содержится $110 + 120 = 230$ калорий. Ограничения задачи заключаются в том, что дети должны получить определенный минимум питательных веществ, а если в каком-либо решении их больше чем

нужно, то это тоже вполне допустимо. В 10 унциях K, например, содержится 1100 калорий, в то время как требуется только 400. Чтобы построить модель линейного программирования, поместим данные нашей задачи в следующую таблицу.

	Содержание в 1 унции K	Содержание в 1 унции C	Потребность
Тиамин	0,10 мг	0,25 мг	1
Ниацин	1,00 мг	0,25 мг	5
Калории	110,00	120,00	400
Стоимость 1 унции	3,8 цента	4,2 цента	

Чтобы выписать ограничения задачи, мы заметим, что общее количество каждого витамина и калорий в левых столбцах должно быть не меньше ($\geqslant$) величины, стоящей в правом столбце. Таким образом, для каждой составной части мы получаем одно ограничение. Например, K унций первого вида содержат $0{,}1\,K$ мг тиамина, а C унций второго вида содержат $0{,}25\,C$ мг этого вещества. Поэтому решение задачи должно удовлетворять неравенству

$$0{,}10K + 0{,}25C \geqslant 1.$$

Аналогично для ниацина получаем

$$1{,}00K + 0{,}25C \geqslant 5,$$

а для калорий

$$110{,}00K + 120{,}00C \geqslant 400.$$

Количество каждого вида пищи должно выражаться положительным числом или нулем, то есть $K \geqslant 0$ и $C \geqslant 0$. Наконец, общая стоимость меню дается формулой

$$3{,}8K + 4{,}2C.$$

Собирая воедино все эти соотношения, мы получаем, что задача нашей хозяйки заключается в следующем: найти величины K и C, минимизирующие общую стоимость

$$3{,}8K + 4{,}2C$$

при условии, что

$$\begin{aligned} 0{,}10K + 0{,}25C &\geqslant 1, \\ 1{,}00K + 0{,}25C &\geqslant 5, \\ 110{,}00K + 120{,}00C &\geqslant 400, \\ K &\geqslant 0, \\ C &\geqslant 0. \end{aligned}$$

Читатель может легко проверить, что два найденных ранее меню $K = 10$, $C = 0$ и $K = 0$, $C = 20$ удовлетворяют данным неравенствам. Минимальное решение получается при $K = 4^4/_9$ унции и $C = 2^2/_9$ унции с общей стоимостью $26^2/_9$ цента. При этом дети получат 1 мг тиамина и 5 мг ниацина, что как раз и требуется, а также $755^5/_9$ калории с излишком в $355^5/_9$ калории [1].

Эту простую задачу о диете, или о выборе меню, можно обобщить. Новая задача состоит в том, чтобы определить суточную диету, которая удовлетворила бы потребности во всех питательных веществах при одновременной минимизации общей стоимости. Подобная задача была впервые сформулирована в начале 40-х годов, еще до открытия методов линейного программирования. Экономист Дж. Стиглер сформулировал задачу о диете, содержавшую 77 видов продуктов питания и использовавшую цены 1939 года. Для решения задачи Стиглер применил метод проб и ошибок. Тщательный анализ и хорошо развитая интуиция позволили ему определить типы и количество продуктов, содержащих необходимый дневной минимум питательных веществ при общей очень низкой, но не минимальной стоимости. В его решение входило только пять продуктов с общей годовой стоимостью 39,93 доллара. Это были пшеничная мука, сухое молоко, капуста, шпинат и сухой горох. Диета минимальной стоимости, полученная с помощью методов линейного программирования, содержит девять продуктов: пшеничную муку, кукурузную муку, сухое молоко, арахисовое масло, лярд, говяжью печень, капусту, картофель и шпинат и дает чуть меньшую годовую стоимость 39,67 доллара. Подобные диеты, хотя и крайне дешевы, но совершенно неудобоваримы и могли бы вызвать восторг разве что у главного диетолога какого-нибудь

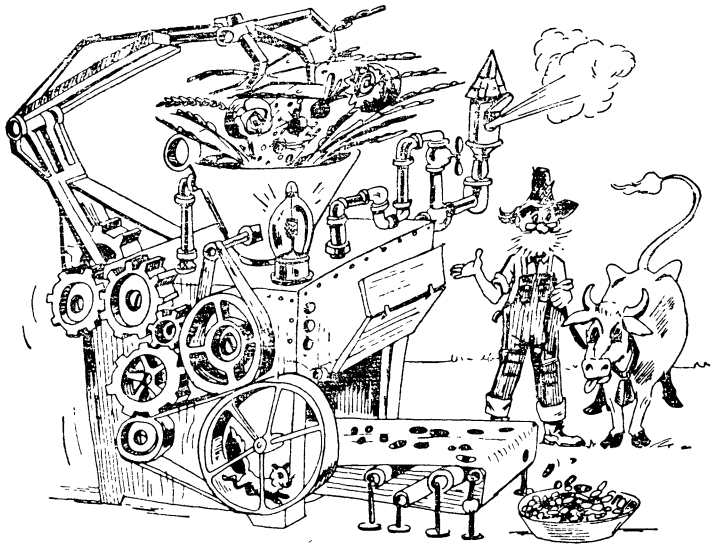
[1] О том, как найдено это решение, рассказано в III главе.

концентрационного лагеря. Сам Стиглер писал: «Никто не собирается рекомендовать подобные диеты (то есть диеты минимальной стоимости) кому бы то ни было, не говоря уж о том, чтобы рекомендовать их всем!» Он привел также диету низкой стоимости, составленную

одним диетологом для 1939 года, общая стоимость которой в год составляла 115 долларов. Разница в стоимости была вызвана тем, что диетолог позаботился уже и о вкусовых качествах и разнообразии диеты, а также учел преимущественную ценность некоторых продуктов. Более поздние попытки составить оптимальную диету с помощью линейного программирования были встречены с пониманием, поскольку аналитики смогли выразить посредством линейных ограничений требования диетологов относительно аромата, вкуса и разнообразия пищи. Подобная разработка меню выполняется теперь для многих организаций и по сравнению с обычным подходом диетологов позволяет разумно экономить средства. Но

даже если успех задачи о составлении рационов для людей и ограничен, аналогичная задача, касающаяся рационов для цыплят, крупного рогатого скота или свиней, представляет собой поучительный пример приложений линейного программирования.

Задача о диете, или задачи о смесях, возникают в самых разных областях. Так, мы встречаемся с ними при составлении самой дешевой смеси кормов для скота или при смешивании различных элементов, например, удобрений или химикалиев, удовлетворяющих по крайней



мере требованиям минимальной стоимости. Математические модели подобных задач представляют собой обобщение той простой задачи, с которой столкнулась наша хозяйка. Даже в случае составления кормов для скота мы встречаем такие ограничения, как вкусовые качества, однако в этом случае с ними легче справиться. Хотя математический аппарат в этих приложениях стандартен, но насколько удастся определить всю совокупность нужных ограничений, зависит от того, хорошо ли разбирается аналитик во всех обстоятельствах, связанных с задачей. Однажды потребовалось составить корм для скота, причем накладывалось ограничение на количество ме-

ляссы[1], поскольку корм должен был изготавливаться на фабрике в виде спрессованных шариков. Выяснилось, что мелясса дешевле, содержит довольно высокий процент кальция и протеина и пришлась коровам по вкусу. Поэтому в оптимальном решении мелассы содержалось столько, сколько было возможно. Тем не менее этот оптимальный корм забраковали. Хотя его и можно было изготовить в форме шариков, высокое содержание мелассы делало навоз столь жидким, что в качестве удобрения он уже не годился. На этом довольно прозаическом примере хорошо видно, как должен подходить аналитик к построению модели для практической задачи. Это эволюционный процесс, и он требует тесного взаимодействия людей и задачи, человека и модели.

ЗАДАЧА ПОСТАВЩИКА

Как и должно было рано или поздно случиться, консультант по вопросам управления получил приглашение на чаепитие к Сумасшедшему Шляпошнику[2], чтобы сделать это чаепитие менее «сумасшедшим». Вот уже много лет Шляпошник, его друзья и гости все пересаживаются и пересаживаются с места на место вокруг стола в тщетных поисках чистых чайных приборов и салфеток, которые давно уже так грязны, что на стол невозможно взглянуть без содрогания.

— Если вы собираетесь по-прежнему принимать гостей каждый день, то вам прежде всего необходимо составить план, — объяснил консультант Шляпошнику.

— Но мне некогда заниматься планированием, — пожаловался Шляпошник, — ведь я должен развлекать моих гостей стихами, загадками и при случае угощать их чаем.

— Тогда предоставьте это мне, — сказал консультант.— Я понаблюдаю за тем, что у вас здесь происходит,

---

[1] Мелясса — кормовая патока, входит в состав кормов для скота. — *Прим. перев.*

[2] Сумасшедший Шляпошник и Мартовский Заяц — персонажи книги Л. Кэролла «Алиса в стране чудес». Часы у Шляпошника всегда показывают пять часов — время чаепития. Поэтому Шляпошник и его гости постоянно пьют чай. Причем сам Шляпошник всякий раз пересаживается на новое, соседнее, место, чтобы пить из чистой чашки, а его гости пересаживаются по кругу вслед за ним, так что им приходится пить из грязных чашек. — *Прим. перев.*

а потом при некотором вашем участии, а также с помощью моей модели мы сделаем так, что чаепития станут приятнее, лучше и дешевле, чем когда-либо раньше.

— Мои чаепития уже многие годы для всех служат образцом. Но мне хотелось бы узнать, что представляют собой ваши. Я нанимаю вас, — сказал Шляпошник.

Гости за столом передвинулись в очередной раз, и наш консультант присоединился к ним с ручкой в одной руке и контрактом в другой. И он и Шляпошник поставили свои подписи под контрактом, что было засвидетельствовано должным образом Мартовским Зайцем. В контракте было указано, что консультанту надлежит изучить данную проблему во всей ее полноте и в течение месяца представить свой отчет и рекомендации.

Ниже мы помещаем этот отчет.

АНАЛИТИЧЕСКОЕ ИССЛЕДОВАНИЕ ВНУТРЕННИХ ВЗАИМОДЕЙСТВИЙ С ТОЧКИ ЗРЕНИЯ ИХ ОТНОШЕНИЯ К ЭКОНОМИКЕ ФУНКЦИОНАЛЬНОЙ СТРУКТУРЫ

Предварительная модель линейного программирования для подсистемы чаепития, составленная консультантом по проблемам управления

Введение

Даже беглое знакомство с операциями чаепития в том виде, в каком они производятся ныне под управлением Сумасшедшего Шляпошника, не оставляет ни малейших сомнений в том, что дела здесь находятся в ужасном состоянии. Любая попытка внести в данные операции некое подобие аналитической методологии привела бы к катастрофе. В соответствии с этим мы рекомендуем приостановить операции чаепития, как только закончится очередной прием гостей, дабы возможно скорее ввести в действие нижеозначенную программу. Несмотря на то что предложенная нами программа затрагивает лишь один аспект операций чаепития, можно быть уверенным в том, что последовательное осуществление программы эффективной стоимости в указываемой нами наиболее важной части операции с высокой вероятностью успеха позволит продвинуть вперед и решение всех других проблем рациональной организации чаепития. Более конкретно мы рекомендуем, чтобы все имеющиеся в наличии грязные салфетки были уничтожены и было введено программирование покупок и стирок новых салфеток*. Мы приступили к этой части общей задачи, используя

* Контрольные эксперименты показали, что салфетки находятся в столь плачевном состоянии, что их отстирать невозможно.

мощные инструменты современной математической науки об управлении. Наш подход состоит в следующем.

Математическая модель

Опираясь на свой богатый опыт в области математического моделирования, мы смогли выделить из всей системы чаепития основную подсистему, допускающую полный математический анализ. Такой анализ основан на хорошо известной задаче поставщика, в которой некий поставщик кулинарных изделий хочет определить, сколько салфеток ему нужно купить, а сколько отправить в прачечную, чтобы постоянно иметь достаточное их количество для своих клиентов. Поставщик стремится достигнуть правильного баланса между покупками и стирками, чтобы минимизировать общую стоимость подсистемы салфеток. Наш план заключается в том, чтобы приложить данную модель линейного программирования к задаче о чаепитии. Оптимизировав поток салфеток, мы сможем распространить свой анализ на другие элементы операций чаепития. Мы убеждены, что через некоторое время нам удастся оптимизировать поток бутербродов, поток гостей и, наконец, поток чая.

Анализ

Дабы проиллюстрировать, как модель, взятую из задачи поставщика, можно применить к нашей задаче, мы вначале занялись первой стадией нашего проекта - сбором данных и определили, что в среднем в течение одной недели ваши чаепития посещает следующее количество гостей:

1) понедельник - 5;
2) вторник - 6;
3) среда - 7;
4) четверг - 8;
5) пятница - 7;

6) суббота - 9;

7) воскресенье - 10.

Каждую покупку новых салфеток можно совершить в любой день и вовремя получить по цене 25 центов за штуку (доставка бесплатная). В округе есть две апробированные и получившие одобрение прачечные. В прачечной Короля салфетку могут выстирать за два дня, причем это обойдется в 15 центов, в то время как в прачечной Королевы ту же работу выполняют за три дня по цене 10 центов за штуку. Допустим, что мы сожгли все старые салфетки, так что их у нас не осталось ни одной штуки, и построим сейчас линейную модель по вашим данным.

Во-первых, введем обозначения. Пусть n_1, n_2, n_3, n_4, n_5, n_6, n_7 - количество новых салфеток, купленных в соответствующий день недели. Аналогично пусть k_1, k_2, k_3, k_4, k_5, k_6, k_7 и q_1, q_2, q_3, q_4, q_5, q_6, q_7 - количество салфеток, отправленных соответственно в прачечные Короля и Королевы. Наконец, пусть d_1, d_2, d_3, d_4, d_5, d_6, d_7 означает количество грязных салфеток, не отправленных в стирку в соответствующий день. Поскольку мы хотим минимизировать общую сумму, необходимую для поддержания количества чистых салфеток на должном уровне, не следует покупать салфеток больше, чем требуется в данный день, или отправлять салфетку в прачечную, если она не будет использована в дальнейшем.

В первый день наших операций, то есть в понедельник, мы должны купить точно такое количество салфеток, какое требуется. Поскольку мы ожидаем пятерых гостей, то

$$n_1 = 5.$$

По окончании чаепития в понедельник мы можем выбирать: послать ли все или некоторые из пяти грязных салфеток в быструю прачечную Короля, в более медленную прачечную Королевы или оставить их в ящике с грязным бельем. То, что происходит с этими пятью салфетками, можно представить уравнением

$$k_1 + q_1 + d_1 = 5$$

Общая стоимость всех операций первого дня равна

$$25n_1 + 15k_1 + 10q_1 \text{ (центов)}.$$

Разумеется, наша задача состоит в том, чтобы точно определить, какие именно числовые значения следует дать нашим переменным (пока что мы рассматриваем n_1, k_1, q_1 и d_1 как переменные, где n_1 равно в точности 5). Как только мы найдем все уравнения нашей задачи, то, воспользовавшись вычислительными методами линейного программирования, сможем получить решение минимальной стоимости. Сейчас мы выпишем остальные ограничения данной модели, помня о том, что в нашей системе выстиранные салфетки возвращаются обратно через 2 или через 3 дня в зависимости от того, какой прачечной мы воспользовались. Для чаепития в среду можно будет использовать k_1 салфеток, отправленных в понедельник, в то время как q_1 салфеток будут готовы к четвергу. Точно так же d_1 салфеток, не отправленных в понедельник, можно отправить на следующий день.

Для чаепития во вторник мы снова должны купить некоторое количество салфеток, а именно

$$n_2 = 6.$$

После того как их используют, мы поступим с ними и d_1 грязными салфетками согласно уравнению

$$k_2 + q_2 + d_2 = 6 + d_1.$$

Оно выражает собой тот факт, что у нас скопилось d_1 грязных салфеток, оставшихся с понедельника, и еще 6, оставшихся со вторника; мы можем либо отдать их в стирку, либо оставить в ящике с грязным бельем. Стоимость всех операций во вторник составляет

$$25n_2 + 15k_2 + 10q_2 \text{ (центов)}.$$

В среду нам требуется 7 салфеток. Это первый день, когда выстиранные салфетки из быстрой прачечной Короля можно снова употребить в дело. Поэто-

му нужные 7 салфеток можно получить, купив или взяв их из того количества, которое было отдано в стирку в понедельник. Мы имеем

$$n_3 + k_1 = 7.$$

Так же как и раньше,

$$k_3 + q_3 + d_3 = 7 + d_2,$$

а стоимость операции в среду равна

$$25n_3 + 15k_3 + 10q_3 \text{ (центам)}.$$

Необходимые в четверг 8 салфеток могут быть либо новыми, либо выстиранными из числа тех, которые мы послали в прачечную Короля во вторник или в прачечную Королевы в понедельник. Это можно записать как

$$n_4 + k_2 + q_1 = 8$$

и

$$k_4 + q_4 + d_4 = 8 + d_3,$$

а стоимость операций в четверг составит

$$25n_4 + 15k_4 + 10q_4 \text{ (центов)}.$$

Мы можем теперь непосредственно выписать оставшиеся уравнения нашей модели. Предположим для простоты, что нас интересует эффективный способ чаепитий только в течение одной недели и, следовательно, мы не будем отдавать салфетки в стирку, если они не вернутся назад к воскресенью.

Для пятницы получаются следующие уравнения:

$$n_5 + k_3 + q_2 = 7,$$

$$k_5 + d_5 = 7 + d_4$$

с общей стоимостью

$$25n_5 + 15k_5 \text{ (центов)};$$

для субботы:

$$n_6 + k_4 + q_3 = 9,$$

$$d_6 = 9 + d_5$$

с общей стоимостью

$$25n_6 \text{ (центов);}$$

для воскресенья:

$$n_7 + k_5 + q_4 = 10,$$
$$d_7 = 10 + d_6$$

с общей стоимостью

$$25n_7 \text{ (центов).}$$

Объединяя полученные выше уравнения, мы видим, что наша задача заключается в том, чтобы найти величины n, k, q и d (эти величины или положительны, или равны нулю, то есть неотрицательны), которые минимизируют функцию стоимости $25(n_1+n_2+n_3+n_4+n_5+n_6+n_7)+15(k_1+k_2+k_3+k_4+k_5)+10(q_1+q_2+q_3+q_4)$ и удовлетворяют линейным уравнениям

$$\begin{aligned}
n_1 &= 5,\\
n_2 &= 6,\\
n_3 + k_1 &= 7,\\
n_4 + k_2 + q_1 &= 8,\\
n_5 + k_3 + q_2 &= 7,\\
n_6 + k_4 + q_3 &= 9,\\
n_7 + k_5 + q_4 &= 10,\\
k_1 + q_1 + d_1 &= 5,\\
k_2 + q_2 + d_2 &= 6 + d_1,\\
k_3 + q_3 + d_3 &= 7 + d_2,\\
k_4 + q_4 + d_4 &= 8 + d_3,\\
k_5 + d_5 &= 7 + d_4,\\
d_6 &= 9 + d_5,\\
d_7 &= 10 + d_6.
\end{aligned}$$

Оптимальное решение

Наши вычисления показывают, что если покупки и стирку производить так, как указано ниже, то чистых салфеток хватит на каждого гостя. Их общая стоимость составит 8 долларов 80 центов, причем, чтобы обслужить 52 ожидаемых гостя, придется купить 21 новую салфетку.

Покупки и стирки за неделю чаепитий

$n_1 = 5$			$d_1 = 0$
$n_2 = 6$			$d_2 = 0$
$n_3 = 7$	$k_1 = 0$		$d_3 = 0$
$n_4 = 3$	$k_2 = 0$	$q_1 = 5$	$d_4 = 0$
$n_5 = 0$	$k_3 = 1$	$q_2 = 6$	$d_5 = 2$
$n_6 = 0$	$k_4 = 3$	$q_3 = 6$	$d_6 = 9$
$n_7 = 0$	$k_5 = 5$	$q_4 = 5$	$d_7 = 10$
21	9	22	21

Общая стоимость в долларах: 21x0,25+9x0,15+22x x0,10=8,80.

Мы убеждены, что математический подход к управлению чаепитиями приводит к огромной экономии средств, а обобщив проделанный выше анализ, его можно применить ко всем операциям Сумасшедшего Шляпошника.

Историческая справка

Задача поставщика впервые появилась в литературе под видом чисто военной задачи. Эта плохо завуалированная попытка скрыть истинное значение и мощь данной модели была быстро преодолена. Для полноты картины приведем оригинальную постановку задачи. Вместо поставщика - военный командир, которому нужны двигатели для самолетов (салфетки) в соответствии с определенными требованиями (число гостей в день). Он может на выбор либо покупать новые двигатели, либо отправлять в ремонт вышедшие из строя. Ремонт бывает срочным (быстрое обслуживание) и обычным, менее дорогим (медленное обслуживание). Конечно, новый двигатель стоит дороже любого вида ремонта.

) Мы приводим этот вид приложения, дабы показать, что применимость многих моделей линейного программирования к различным задачам зависит от изобретательности аналитика. Можно, например, показать, что задача поставщика на самом деле представляет собой замаскированную транспортную задачу.

ЗАДАЧА ОБ ОПТИМАЛЬНОМ РАСПОЛОЖЕНИИ

То был немилосердный удар,
нанесший глубокую рану.

Обычно каждый студент, изучающий курс исследования операций в соответствии с программой, посещает какую-либо развитую организацию (промышленную или иную), чтобы ознакомиться с одной из сторон ее деятельности. Среди отчетов о проделанной за семестр работе можно встретить такие фундаментальные труды, как «Применение математических методов управления к сбору макулатуры», «О порядке очередей в универмаге» или «Контроль уличного движения на воде: исследования транспортного потока в Венеции, Италия».

На долю одной группы студентов досталась высокоразвитая химическая компания. В такого рода организации всегда много операционных задач, связанных с планированием производства, хранением, транспортировкой и т. п. Побывав на нескольких заводах этой компании, студенты решили исследовать операции, связанные с производством целлофана.

Вискоза, которая образуется при обработке древесной пульпы щелочью и другими химическими веществами, проталкивается сквозь длинные узкие щели в чан с серной кислотой. Здесь она, мгновенно превращаясь в целлюлозу, приобретает форму широкой ленты. Эта лента проходит через ряд других чанов, где ее очищают, промывают, сушат на барабанах и, наконец, сворачивают (подобно ковру) в рулоны.

Затем рулоны отправляются на склад, где хранятся до тех пор, пока их не разрежут на части меньшей ширины (в соответствии с поступившими заказами). Каждый большой рулон имеет ширину 60 дюймов. Когда такой рулон требуется разрезать на рулоны меньших размеров, его отправляют в специальное помещение и поднимают на резальную машину. Этот механизм разворачивает полутораметровую ленту, и одновременно специально установленные лезвия режут целлофан на куски меньшей ширины, которые свертываются в рулоны на другом конце машины. При этом стремятся, кроме рулона заказанной ширины, и остальные рулоны получить таких размеров, на которые может быть спрос. Например, 60-дюймовый рулон можно разрезать на три

15-дюймовых рулона, 10-дюймовый и 5-дюймовый рулоны. Однако если 5-дюймовый рулон никто не заказывает, он идет в отходы и составляет потерю. При каждом расположении лезвий (в нашем случае понадобились четыре лезвия) получаются рулоны, которые либо продаются, либо составляют потери.

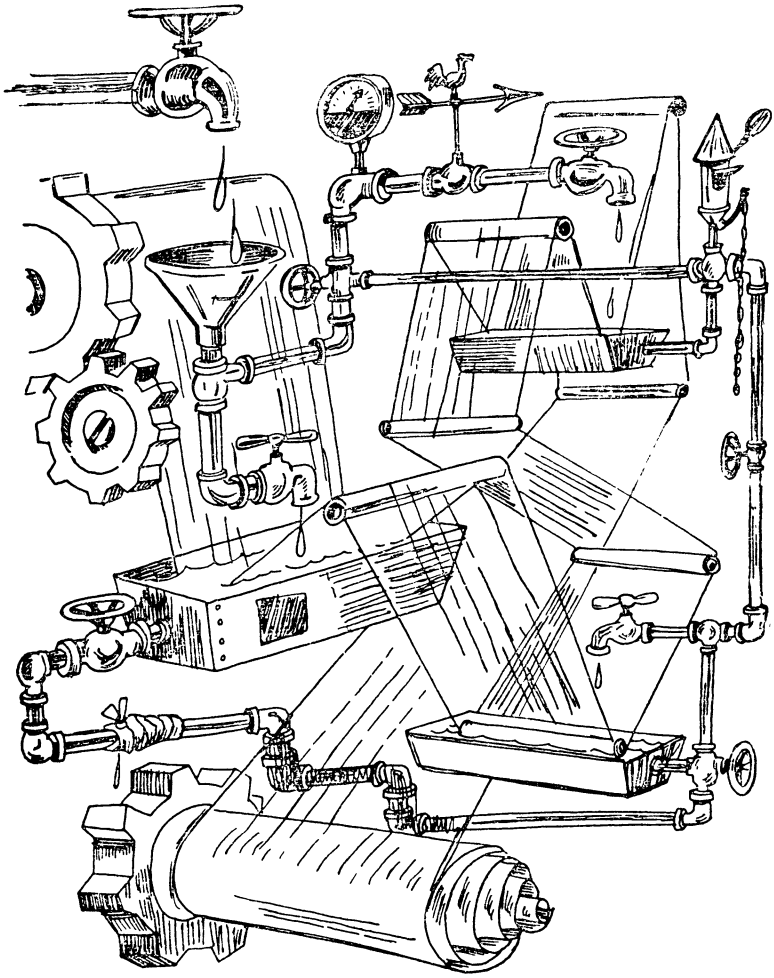
Собрав все недельные заказы на рулоны различной ширины, мастер пытается сгруппировать их таким образом, чтобы получить требуемые рулоны, одновременно минимизировав потери. Опытному мастеру удается так скомбинировать заказы, чтобы потери были достаточно низкими.

Во время экскурсии по заводу студенты заметили, как рвут на части и упаковывают целые кипы отходов. А нельзя ли сэкономить на этой расточительной и дорогостоящей части операций? Студенты довольно быстро поняли, что в действительности управление процессом основано на опыте мастера и попытались оценить количественно его эвристический подход к оптимизации. Опираясь на работу, первоначально проделанную одной канадской компанией, наша группа предложила испробовать модель, которая уже дала существенную экономию в бумажной промышленности. Студенты решили показать преимущества модели, сравнив решения мастера с решениями линейной программы для тех же недельных данных. В подобных ситуациях, когда новомодная математика «подкапывается» под человеческий опыт, психологически и стратегически очень важно добиться сотрудничества и взаимодействия с заинтересованной стороной. У членов группы, хотя они и были новичками в этом деле, оказалось достаточно смекалки, чтобы подружиться с мастером и вовлечь его в свой эксперимент (так им, во всяком случае, казалось).

Мастер сообщил им следующие данные о заказах текущей недели:

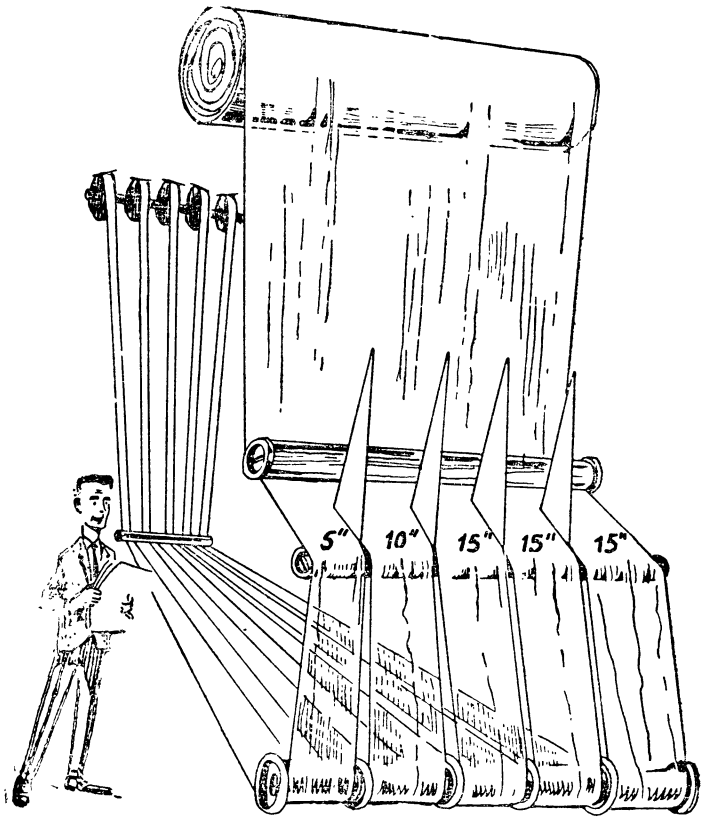
Требуемая ширина в дюймах	Требуемое количество рулонов
28	30
20	60
15	48

Все это требовалось нарезать из стандартных 60-дюймовых рулонов. Предполагалось, что количество широких рулонов достаточно для того, чтобы удовлетворить все недельные заказы. Учитывая тип целлофана, произ-



водимого на этой неделе, всякий оставшийся рулон шириной менее 15 дюймов следовало рассматривать как потерю.

Ключом к построению соответствующей линейной модели послужил удачный выбор переменных. При каж-



дом расположении режущих лезвий получался определенный набор небольших рулонов, и задача, следовательно, состояла в том, чтобы определить, как следует расположить лезвия и сколько больших рулонов нужно разрезать при данном их расположении. Например, для того чтобы получить 28-дюймовые рулоны, можно исполь-

зовать два лезвия и разрезать 60-дюймовый рулон на три куска — два рулона шириной по 28-дюймов и один рулон шириной 4 дюйма. Первые рулоны пойдут на заказы, а последний рулон составит потерю. Таким образом, прежде всего нужно определить, при каком расположении лезвий получаются рулоны, которые можно использовать. Каждое отдельное расположение представляет собой переменную нашей задачи, а значение каждой переменной равно числу 60-дюймовых рулонов, которые следует разрезать при данном расположении лезвий. Пусть x_1 показывает, сколько раз 60-дюймовые рулоны разрезаются на два 28-дюймовых рулона с 40-дюймовой потерей, и пусть другие допустимые расположения задаются следующей таблицей:

Требуемая ширина в дюймах	x_1	x_2	x_3	x_4	x_5	x_6	x_7	Число заказанных рулонов
28	2	1	1	0	0	0	0	30
20	0	1	0	3	2	1	0	60
15	0	0	2	0	1	2	4	48
Потери	4	12	2	0	5	10	0	

Переменная x_3 показывает, сколько раз 60-дюймовые рулоны разрезаются на один 28-дюймовый рулон и два 15-дюймовых рулона с 2-дюймовой потерей. Остальные переменные определяются аналогично. Каждая переменная неотрицательна ($\geqslant 0$) — мы или разрезаем рулоны при данном расположении лезвий, или не разрезаем. Теперь уже довольно легко построить математическую модель.

Составляя уравнения, студенты выяснили, что ради большей гибкости и свободы управления разрешается нарезать больше рулонов нужной ширины, чем заказано. Всякий излишек можно отправить на склад, чтобы использовать при удовлетворении заказов на следующей неделе. Если бы это было не так, пришлось бы нарезать ровно тридцать 28-дюймовых рулонов, шестьдесят 20-дюймовых рулонов и сорок восемь 15-дюймовых рулонов. В нашем же случае ограничения позволяют про-

изводить большее количество рулонов, чем требуется, и поэтому задаются неравенствами вместо равенств.

Составив приведенную выше таблицу, студенты смогли выписать три главных ограничения линейной модели — по одному ограничению для каждой ширины. Для 28-дюймовых рулонов они имели

$$2x_1 + x_2 + x_3 \geqslant 30,$$

для 20-дюймовых —

$$x_2 + 3x_4 + 2x_5 + x_6 \geqslant 60$$

и для 15-дюймовых —

$$2x_3 + x_5 + 2x_6 + 4x_7 \geqslant 48.$$

Кроме того, все x должны были быть больше или равны нулю ($\geqslant 0$). Наконец, общие потери давались выражением

$$4x_1 + 12x_2 + 2x_3 + 0 \cdot x_4 + 5x_5 + 10x_6 + 0 \cdot x_7.$$

Студентам требовалось найти множество x, удовлетворяющих данным неравенствам и минимизирующих общую меру потерь.

Одним из решений, не обязательно минимальным, было то, которым пользовался мастер. Но существовали и другие решения. Если $x_1 = 15$, $x_4 = 20$ и $x_7 = 12$, а все остальные x равны нулю, то требуемые ограничения выполняются (причем как равенства) и общие потери составят $4x_1 + 0 \cdot x_4 + 0 \cdot x_7 = 4 \cdot 15$(дюймов). Это решение означает, что нужно разрезать 15 широких рулонов при расположении лезвий x_1 (нарезаются два 28-дюймовых и один 4-дюймовый рулон), 20 широких рулонов — при расположении x_4 (нарезаются три 20-дюймовых рулона) и 12 широких рулонов — при расположении x_7 (четыре 15-дюймовых рулона); при этом получится только 60 дюймов потерь. А существуют ли решения с меньшими потерями?

Заметим, что если взять $x_3 = 30$, $x_4 = 20$ и $x_7 = 12$, то мы снова удовлетворим всем ограничениям (15-дюймовых рулонов будет больше, чем нужно), а потери снова составят 60-дюймов. Мы предоставим читателю показать, что в каждом из этих двух решений потери минимальны.

Получив свое решение, студенты вместе с мастером попытались сравнить его с решением мастера. Однако мастер заметил, что такое сравнение невозможно и что при построении модели ребята слишком абстрагировались от реальной задачи. Он попытался объяснить, в чем состояла их ошибка.

Решая, как следует разрезать каждый рулон, нельзя не учитывать его физических характеристик. В отличие от бумажной промышленности, где рулоны практически изготовляются без дефектов, на заводе, куда попали студенты, процесс свертывания целлофана был весьма несовершенен. Большинство рулонов выглядело подобно свернутым коврам: на одном конце рулона — воронка, на другом — выступ. Такой рулон очень редко можно использовать целиком. Как правило, концы его приходится отрезать, так что в дело он идет уже в более узком виде. Кроме того, нередко брак встречается и внутри рулонов, так как при свертывании целлофана трудно поддерживать правильное натяжение. Бракованные места приходится вырезать подобно тому, как это делают с дырками от сучков в досках. Это тоже вызывает лишние потери. Таким образом, львиная доля потерь обязана своим появлением процессу производства, а не процессу резания.

Выяснив все эти моменты, студенты снова приступили к работе. Уточнили, как образуются потери (дефекты в рулонах, брак при разрезании) и обнаружили, что это происходит не всегда по вине мастера. Когда мастер находит в рулоне бракованное место, он должен приспособиться к новой ситуации и так перегруппировать заказы, чтобы как можно рациональнее использовать нестандартный рулон.

Студенты описали свой опыт построения модели вместе с рекомендацией детально изучить процесс резки. Они указали и возможный путь решения проблемы: резальную машину следует оснастить электронным устройством, которое будет находить бракованные места в рулонах и передавать эти данные вычислительной машине. Затем, используя методы линейного программирования, студенты построили модель, с помощью которой можно определить расположение лезвий для любого рулона с учетом имеющихся заказов, причем лезвия должны автоматически устанавливаться вычислительной машиной. Студенты произвели предварительную оценку за-

трат и сравнили эффективность действующего в настоящее время и предложенного методов. Группа закончила свой отчет настоятельной рекомендацией, чтобы мастер оставался на своем месте.

Замечено, что человек, хорошо тренированный в решении не слишком объемистых оптимальных задач, легко справляется с ними и получает решения, довольно близкие к оптимальным. Именно так обстояло дело со Стиглером в задаче о диете и с нашим мастером. По мере усложнения задач начинает преобладать роль математической модели, а роль человека ей заметно уступает. Но так случается тоже не всегда.

ЗАДАЧА О НАЗНАЧЕНИИ ПЕРСОНАЛА

Еще сравнительно недавно новобранцу предлагали множество тестов, дабы определить, в каком роде войск он принесет больше пользы. С помощью тестирования определяли его способности к механике, электро- и радиотехнике, подсчитывали баллы, изучали потребности, и он оказывался... в пехоте.

Сегодня дело обстоит несколько иначе. Тесты все еще используются, потребности еще анализируются, но уже есть надежда, что новобранец закончит свою службу на том месте, которое требует качеств, имеющих некое отношение к его истинным способностям. Проблема, состоящая в том, чтобы правильно распределить наличные людские ресурсы в соответствии с профессиональными требованиями, изучается как в армии, так и в промышленности. И именно линейное программирование является важным инструментом в области рациональной классификации персонала. Центральное место в этих исследованиях занимает задача о назначении персонала, которая может быть поставлена и решена с помощью модели линейного программирования.

Заглянем в один из центров по приему новобранцев в тот момент, когда там царит затишье. Присутствуют только три новобранца, которых мы по вполне понятным причинам назовем Эйбл, Бэйкер и Чарли[1]. Молодым людям была предложена серия тестов, чтобы определить их способности к профессиям радиста,

[1] По-английски эти имена начинаются с первых трех букв латинского алфавита — Able, Baker, Charlie. — *Прим. перев.*

программиста и писаря. Полученные ими баллы представлены в следующей таблице:

	Радист	Программист	Писарь
Эйбл	5	4	7
Бэйкер	6	7	3
Чарли	8	11	2

Чем выше балл, тем выше способности новобранца к данной профессии. Из Чарли, например, получился бы, вероятно, хороший программист, но никудышный писарь. Центр имеет в своем распоряжении трех человек (наших Эйбла, Бэйкера и Чарли) и должен направить одного из них в школу радистов, другого — в школу программистов, а третьего — в школу писарей. Центр должен решить, кого куда направить с максимальной пользой для армии.

Подход к этой задаче, использующий линейное программирование и психологические тесты, основан на допущении, что полученные при испытании баллы дают нам меру ценности человека для соответствующей работы и что общая ценность всех назначений равна сумме баллов. Так, например, если Эйбла направляют на первую работу, Бэйкера — на вторую и Чарли — на третью, как это показано в таблице, то общий балл равен $5 + 7 + 2 = 14$. Можно было бы возразить против подобного способа измерения общей ценности (мы здесь приняли, что соотношения между ценностями индивидуальных назначений линейны). Читателю, который обеспокоен тем, что здесь используются подобные психологические измерения, мы предлагаем ознакомиться с литературой по данному вопросу.

	Радист	Программист	Писарь
Эйбл	1	0	0
Бэйкер	0	1	0
Чарли	0	0	1

Приведенная выше таблица называется таблицей назначений, так как она устанавливает назначение людей на работу (каждый человек назначается на работу и на каждую работу кто-то назначается). Итак, таблица назначений представляет собой квадратную табличку с точно одной единицей в каждой строке и столбце. Другой способ назначения для наших новобранцев задается следующим образом:

	Радист	Программист	Писарь
Эйбл	0	1	0
Бэйкер	1	0	0
Чарли	0	0	1

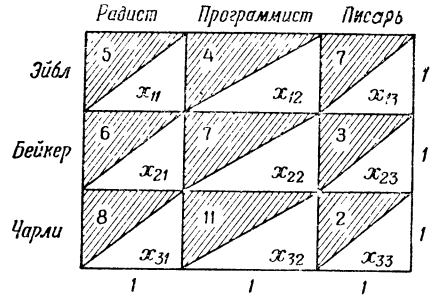
Ценность этого способа равна $6 + 4 + 2 = 12$. Для нашей задачи (3×3) существует только $3 \times 2 \times 1 = 6$ возможных способов. Легко перебрать все возможности и показать, что способ

	Радист	Программист	Писарь
Эйбл	0	0	1
Бэйкер	1	0	0
Чарли	0	1	0

дает максимальный балл, равный $6 + 11 + 7 = 24$. Когда число людей достаточно велико, такой метод перебора становится непрактичным. Однако линейную модель в этом случае можно построить непосредственно.

Здесь было бы полезно рассмотреть эту задачу в свете того, что мы уже знаем о транспортных задачах. По существу, у нас есть несколько пунктов отправления, называемых Эйбл, Бэйкер и Чарли. Из каждого такого пункта в некоторые пункты назначения требуется перевезти одну единицу материала (новобранца). Каждому пункту назначения (радист, программист и писарь)

требуется одна из трех, имеющихся в наличии единиц. В отличие от транспортной задачи мы хотим так организовать перевозку, чтобы максимизировать общие расходы (в действительности — сумму баллов, соответствующих данным назначениям). Поместив все данные в таблицу вроде той, которую мы строили для транспортной задачи, мы получим



Переменные интерпретируются как назначение соответствующего человека на соответствующую работу, то есть x_{23}, например, означает, что Бэйкер направлен в школу писарей. Каждая переменная должна быть неотрицательной; фактически она может принимать только значения, равные единице или нулю. Приведенное выше оптимальное решение устанавливает, что $x_{21}=1$, $x_{32}=1$, $x_{13}=1$, а все другие x равны нулю. Вспоминая, что мы делали в случае транспортной задачи, получим следующую линейную модель. Найти неотрицательные значения переменных $x_{ij} \geqslant 0$, которые максимизируют

$$5x_{11}+4x_{12}+7x_{13}+6x_{21}+7x_{22}+3x_{23}+8x_{31}+11x_{32}+2x_{33}$$

при условии, что выполняются ограничения

$$\begin{aligned}
x_{11}+x_{12}+x_{13} &= 1,\\
x_{21}+x_{22}+x_{23} &= 1,\\
x_{31}+x_{32}+x_{33} &= 1,\\
x_{11}+x_{21}+x_{31} &= 1,\\
x_{12}+x_{22}+x_{32} &= 1,\\
x_{13}+x_{23}+x_{33} &= 1.
\end{aligned}$$

Совершенно ясно, что в решении данной задачи переменные должны принимать целые значения (ноль или

единицу). К счастью, математическая структура этой задачи та же, что и транспортной, поэтому есть гарантия того, что существует оптимальное решение в целых числах. В задаче об оптимальном расположении мы не упоминали о подобном требовании. Там модель могла привести к решению, при котором данное расположение должно использоваться в течение дробной доли времени. Но в производственных задачах не будет никакого вреда, если вдруг появятся дроби. Это будет просто означать, что на самом деле стоимость получится немного выше минимальной. Когда же речь идет о числе людей, то мы обязаны найти решение в целых числах.

Бывает, что в задачах о назначении требуется минимизировать, а не максимизировать целевую функцию. Например, если бы мы занимались перемещением людей на другую работу и «тестовая ценность» представляла бы собой время, необходимое данному человеку для того, чтобы переменить профессию, то мы постарались бы сделать такие новые назначения, при которых общее время было бы минимальным. В любом случае с помощью одних и тех же вычислительных приемов линейного программирования мы можем решить оптимальные задачи всех типов.

Интересный вариант задачи о назначении (который показывает, что линейное программирование важно и для социологии) представляет собой задача о браке. Пусть, например, мы имеем 100 мужчин и 100 женщин и хотим разбить их на пары так, чтобы общее «счастье» данного разбиения было максимальным. Нам нужно составить числовую таблицу, которая показывает «счастье» каждого из ста возможных способов, которыми данный мужчина может выбрать какую-либо женщину. Если мы обозначим через x_{ij} ту часть времени, которую i-й мужчина проводит с j-й женщиной, и если наша цель состоит в том, чтобы максимизировать общее «счастье», то математическая модель нашей задачи просто совпадет с аналогичной моделью громоздкой задачи о назначении персонала. Как уже отмечалось, оптимальные решения таких моделей выражаются в целых числах (здесь 0 или 1). Таким образом, хотя каждый мужчина может поделить свое время между несколькими женщинами, для общественного «счастья» лучше, если он не будет волокитой. Но, как сказал кто-то из репортеров Дж. Б. Данцигу, одному из создателей линейного

программирования, впервые сформулировавшему эту задачу, мы вполне удовлетворились бы и неоптимальными решениями.

ЗАДАЧА О ПЛАНИРОВАНИИ ПРОИЗВОДСТВА

Вы слыхали или нет?
Был построен драндулет:
В день проходит сотню лет.

Современный подход к производственным операциям некоторого предприятия состоит в том, что каждый вид продукции описывается в терминах количества ресурсов, необходимых для производства единицы данного вида продукции. Само производство данной единицы рассматривается как совместное использование некоторых наличных ресурсов. Перед руководителем предприятия стоит следующая задача: определить, сколько нужно произвести единиц каждого вида (то есть определить уровень деятельности), чтобы компания получила максимальный доход при наличии ограничений, связанных с количеством ресурсов, имеющихся в его распоряжении. Этот довольно упрощенный подход (описание комплексных операций в терминах основных взаимосвязанных факторов производства) представляет собой тем не менее мощный метод исследования широкого круга производственных процессов. Его неотъемлемую часть составляет линейная модель. Чтобы познакомиться с этой моделью, давайте заглянем на ежемесячное заседание директоров Мебельной Компании Простофиль: наш девиз *Proba mers facile emptorem reperit* [1]. Президент Саймон [2] как раз начинает рассказывать о новом подходе его компании к технологии производства.

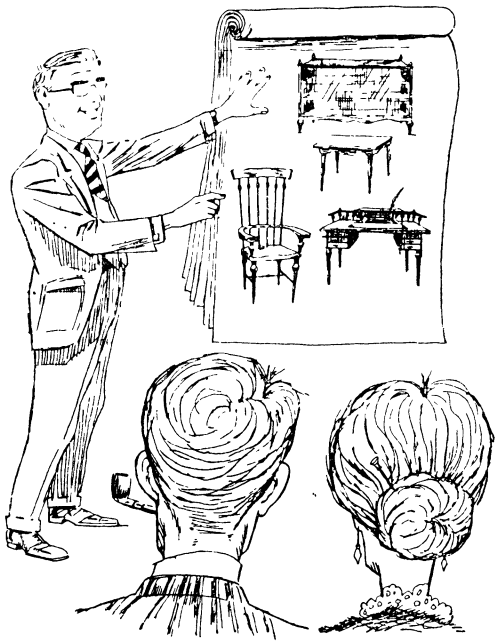
— Джентльмены и, разумеется, матушка Саймон! Недавнее исследование организации нашего производства, проведенное консультантами по вопросам управления, пролило свет на то, какими путями можно улучшить деятельность нашей компании. Чтобы за то короткое время, которое мне предоставлено, пояснить суще-

[1] Хороший товар легко находит покупателя (*лат.*).

[2] Simple Simon (*англ.*) — Простофиля Саймон — персонаж, примерно соответствующий нашему Иванушке-дурачку. Все руководящие должности в Мебельной Компании Простофиль занимают братья Саймон. — *Прим. перев.*

ство этого нового подхода, я позволю себе предельно упростить предмет обсуждения. Многое из того, о чем я буду упоминать, изображено вот на этих плакатах.

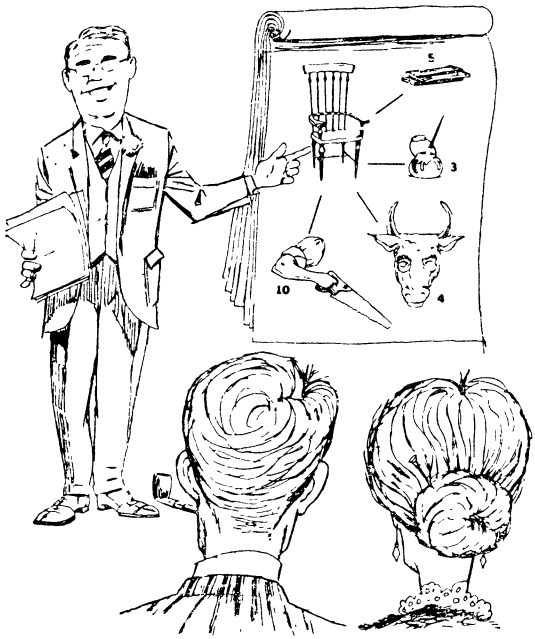
На первом плакате я показал четыре предмета той мебели, которую мы изготовляем из настоящего красного дерева: стул, стол, письменный стол, книжный шкаф. Вы, конечно, заметили, что, кроме основного ингредиента, красного дерева, для производства некоторых из



этих предметов использована кожа; нам нужны стеклянные дверцы для книжного шкафа; во всех случаях требуется клей и, кроме того, некоторые мелкие предметы вроде шурупов, которыми мы пренебрежем. Когда я упомянул слово «стул», нам всем представилось одно из наших восхитительных изделий, стоящих на полу образцово-показательной комнаты. Однако с точки зрения производства картина выглядит иначе — примерно так, как это изображено на следующем плакате.

Вы видите, что один стул в действительности эквивалентен 5 футам досок красного дерева, плюс 10 чело-

веко-часам работы, 3 унциям клея и 4 квадратным футам кожи. Мы просто собираем вместе эти ресурсы, чтобы они стали похожи на стул. Стол эквивалентен 20 футам досок, 15-человеко-часам и 8 унциям клея. Наша радость и гордость — письменный стол Саймона — это на самом деле 15 футов досок, 25-человеко-часов,



15 унций клея и 20 квадратных футов кожи, и он прекрасен! Книжный шкаф требует 22 фута досок, 20 человеко-часов, 10 унций клея и 20 квадратных футов стекла.

Возьмем среднюю продукцию за неделю. В начале недели наш вице-президент по вопросам производства Сид Саймон получает перечень всех наличных ресурсов, производит быстрый подсчет и сообщает своим парням, сколько нужно сделать единиц мебели каждого наименования. Моя цель состоит совсем не в том, чтобы лишить Сиднея работы, я просто хочу дать ему научные

рекомендации, с помощью которых он сможет увеличить доходы. Вот на этом плакате я поместил общие наличные ресурсы на неделю.

20 000 футов досок красного дерева
4 000 человеко-часов
2 000 унций клея
3 000 квадратных футов кожи
500 квадратных футов стекла

Задача Сида определить, как надо распорядиться всеми этими ресурсами, чтобы извлечь максимальный доход. Для этого он справляется у вице-президента по финансовым вопросам Гарри Саймона о последних данных относительно дохода, извлекаемого от производства каждой единицы продукции. Их я изобразил здесь:

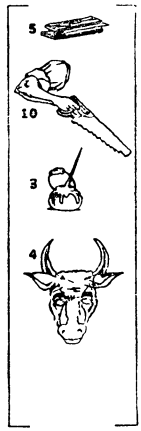
45 долларов за стул
80 долларов за стол
110 долларов за письменный стол
55 долларов за книжный шкаф

Сколько единиц должны мы сделать? Каким образом мы можем получить максимальную прибыль? Правильно ли поступает Сид? Разрешите мне показать вам, как наша новая модель линейного программирования отвечает на эти и другие вопросы.

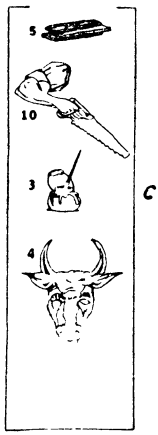
Мне придется привлечь математику, но я попытаюсь пользоваться ею как можно меньше. Ведь все мы давно уже не ходим в школу! Я хочу обозначить число единиц каждого вида мебели, которое необходимо произвести, его начальными буквами. Соответственно число стульев будет c, число столов t и т. д.[1] Если $c = 10$, то я сде-

[1] По-английски стул — chair, стол — table, письменный стол — desk, книжный шкаф — book-case. Поэтому соответствующими обозначениями будут c, t, d, b. — *Прим. перев.*

лаю десять стульев. Для каждого стула я использую следующие ресурсы:

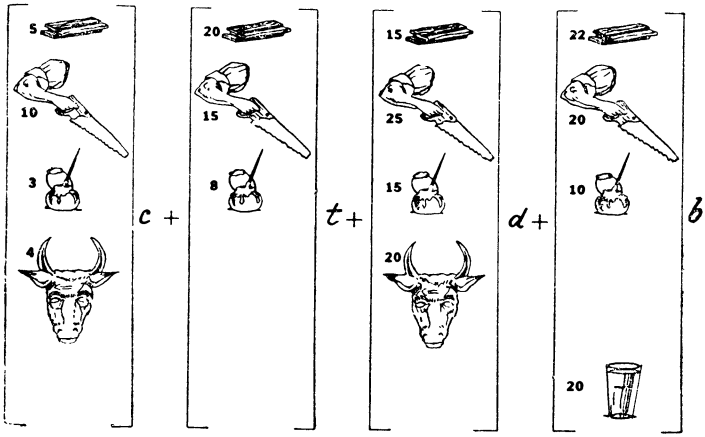


Если я сделаю *c* стульев, то использую такие ресурсы *c* раз, или

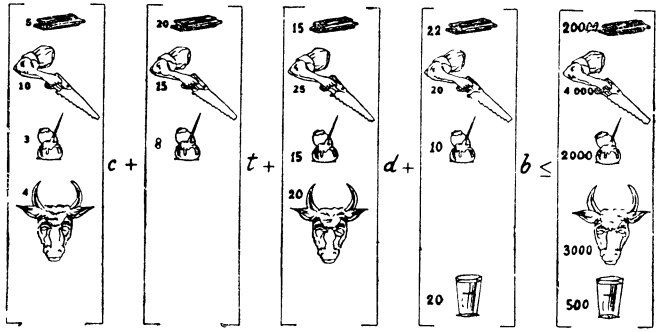


То же самое верно для столов, письменных столов и книжных шкафов. Я могу представить общее количество ресурсов, пошедших на производство *c* стульев, *t* сто-

лов, d письменных столов и b книжных шкафов, следующим образом:



Это общее количество должно быть меньше или равно количеству наличных ресурсов, так что я буду иметь



Доход представляется следующим выражением:

$$45c + 80t + 110d + 55b \text{ (долларов)}.$$

Нам остается теперь найти правильные значения c, t, d, b, которые сделают значение суммы наибольшим. Это как раз и пытается сделать Сид. Линейное программирование гарантирует нам максимальный доход. Вопросы есть? Пожалуйста, Сид.

— Ты не коснулся многих вопросов. Я уверен, что все пойдет с молотка. Парни, которые бывают здесь проездом, заходят к нам и делают заказы, причем они хотят, чтобы их заказы были выполнены без промедления. А при таком подходе, о котором ты здесь разливался соловьем, у меня нет гарантии, что я, например, вообще сделаю хоть один стул. Точно так же я могу сделать слишком много стульев или даже только одни стулья. Как ты позаботишься об этом?

— Очень просто, Сид. Разреши мне написать несколько неравенств.

$$\begin{aligned} 5c + 20t + 15d + 22b &\leqslant 20\,000, \\ 10c + 15t + 25d + 20b &\leqslant 4000, \\ 3c + 8t + 15d + 10b &\leqslant 2000, \\ 4c + 20d &\leqslant 3000, \\ 20b &\leqslant 500. \end{aligned}$$

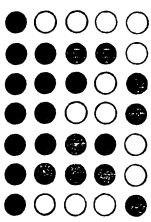
Если тебе нужно сделать как минимум 50 стульев, то надо всего лишь добавить к этой модели одно неравенство

$$c \geqslant 50.$$

Если ты не хочешь делать более 30 письменных столов, то тебе надо сюда добавить

$$d \leqslant 30.$$

Все это очень просто. Наша модель линейного программирования может позаботиться обо всем этом и о многом другом. Мы можем включить сюда задачи о транспортировке, инвентаре и хранении и попытаться оптимизировать всю Простофильную Систему. Этим мы займемся в ближайшем будущем. Вопросов больше нет? Мое время истекло.



РЕШЕНИЕ ЗАДАЧ

(Мы комбинируем некоторые идеи Евклида с интуицией двадцатого века, чтобы понять процессы, лежащие в основе решения задач линейного программирования.)

Те читатели, которые интересуются исключительно решением какой-то конкретной задачи линейного программирования, могут спокойно пропустить эту главу и сразу перейти к следующей. Практически мы в состоянии решить почти любую задачу линейного программирования. Существуют вычислительные методы, которые позволяют решать «вручную» не очень громоздкие задачи. Даже в случае больших (и не очень больших) задач те же самые вычислительные методы вместе с электронно-вычислительными машинами составляют тот материал, из которого куется главное оружие аналитика. Первое, чему «обучаются» вычислительные машины после того, как они освоят основные арифметические действия, — это решение задач линейного программирования. Этот факт свидетельствует о всепроникающей силе линейной модели [1].

Я надеюсь, что не многие читатели пройдут мимо основного содержания этой главы. Хотя и можно разбираться в линейном программировании, не зная толком, как решать задачу (и даже применять такие куцые знания на практике), все же аксиомой является утверждение, что настоящие глубокие знания в этой области приобретаются только при тщательном изучении теоретического, вычислительного и прикладного аспектов. Поскольку я сознательно игнорирую часть этого предписания (теоретическую стену всего здания), похоже, что «здание» и главная цель данной книги не

[1] Поскольку начало развития электронно-вычислительной техники и линейного программирования приходится приблизительно на одно и то же время (конец 40-х годов) и эти две области оказали благотворное влияние друг на друга, весьма некстати произошла серьезная путаница в терминологии. Когда мы говорим о программировании, то не всегда ясно, идет ли речь о логическом анализе задачи или машинных командах, необходимых для ее решения. Так, например, существует программа для решения задач линейного программирования.

выдержат проверки без некоторого обсуждения вычислительной стороны дела. Поэтому я немного поговорю о вычислительном процессе, связанном с линейным программированием, чтобы читатель смог глубже проникнуть в существо дела и лучше в нем разобраться.

Для того чтобы облегчить себе работу, мы должны ввести стандартные математические обозначения. Нам придется, вообще говоря, иметь дело с очень простыми числовыми примерами, в математических моделях которых используются только две переменные. Мы обозначим эти переменные символами x_1 и x_2. С точки зрения планиметрии (евклидовой) x_1 представляет собой одно измерение, или направление, а x_2 — другое измерение, или направление. Выражаясь математическим языком, мы имеем дело с двумерным евклидовым пространством. Геометрия не только поможет нам изобразить это пространство с помощью двух обычных осей координат, но и поможет при решении задач (подробней об этом позже).

В примерах второй главы переменные в зависимости от их определения в конкретной задаче имели различный смысл. Например, в задаче об оптимальном расположении иксы (x) представляли собой расположения режущих лезвий. В некоторых задачах переменные обозначали определенные названия, вроде K в задаче о диете. Обозначим теперь все переменные только буквой x с подходящим числовым или буквенным индексом. Переменную общего вида обозначим символом x_j. Поскольку мы будем решать задачи, представляющие собой просто числовые и символические «игрушечные» примеры, нам придется привыкнуть к обращению с переменными, не имеющими конкретного смысла. Так, например, мы можем говорить об уравнении $x_1 + x_2 = 1$, не утруждая себя интерпретацией этих переменных.

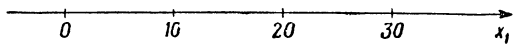
Хотя в большинстве примеров, обсуждаемых в этой главе, используются только две переменные, читателю не составит труда распространить соответствующие понятия и обозначения на ситуации, в которых требуется большее число переменных. В задаче об оптимальном расположении потребовалось 7 переменных, или измерений, однако при ее формулировке нам не пришлось расширять свой опыт и интуицию, подобно тому как это приходится делать в теории относительности. Пространство, в которое нам предстоит вступить, доступно всем.

ОДНОМЕРНЫЕ ЗАДАЧИ

(Для тех, кто впервые знакомится с предметом.)

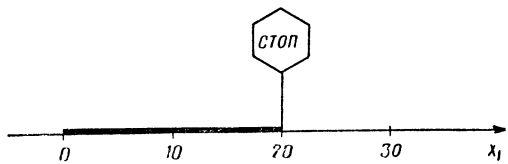
Расстояние — ничто; действительно, важен лишь первый шаг.

Простейшая из всех оптимальных задач состоит в том, чтобы найти максимальное значение одной переменной x_1, когда на нее не накладывается никаких ограничений. Здесь числовое значение x_1 может стать столь большим, сколь нам заблагорассудится, и мы скажем, что максимальное значение неограниченно велико. Геометрически мы ищем наибольшее значение x_1, двигаясь вдоль числовой прямой, или оси.



Поскольку на x_1 нет никаких ограничений, мы можем двигаться вдоль этой прямой сколь угодно далеко вправо, не встречая на пути никаких барьеров. Если на x_1 наложить ограничения, то задача станет более интересной, но все еще очень легкой. Мы будем иметь дело с переменными, значения которых подчинены основному требованию линейного программирования — требованию неотрицательности, следовательно, $x_1 \geqslant 0$.

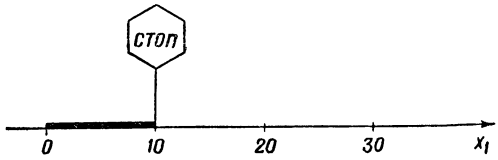
В задаче об отыскании максимального x_1, удовлетворяющего неравенству $x_1 \leqslant 20$, мы можем двигаться вдоль оси x_1 от точки $x_1 = 0$ до тех пор, пока не натолкнемся на барьер (стоп-сигнал) в точке $x_1 = 20$.



Всякая точка на прямой между $x_1 = 0$ и $x_1 = 20$ представляет собой потенциальное решение нашей задачи. Это множество точек называется множеством решений, или пространством решений. Поскольку мы ищем максимум, оптимальной точкой в пространстве решений нашей задачи (точкой, в которой целевая функция достигает своего наибольшего значения) будет $x_1 = 20$. Если

бы мы искали минимальное x_1, удовлетворяющее неравенству $x_1 \leqslant 20$, то оптимальным ответом (помня о требовании неотрицательности) был бы $x_1 = 0$.

Заменив неравенство в задаче о максимуме на $2x_1 \leqslant 20$, мы ограничим свою свободу передвижения вдоль оси x_1 участком, лежащим между точками $x_1 = 0$ и $x_1 = 10$, а оптимальным решением будет $x_1 = 10$. Од-



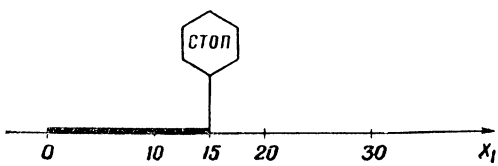
номерные задачи с одним ограничением можно легко решать таким путем. Если единственное ограничение представляет собой уравнение вроде $x_1 = 20$ или $2x_1 = 20$, то пространство решений состоит всего из одной точки, в нашем случае $x_1 = 20$ и $x_1 = 10$ соответственно. Задачи линейного программирования могут включать в себя ограничения, заданные как в виде уравнений, так и в виде неравенств, однако геометрическая природа таких задач станет отчетливей, если мы уделим основное внимание неравенствам.

Типичным примером одномерной задачи линейного программирования со многими ограничениями служит задача об отыскании максимального x_1 при условии, что

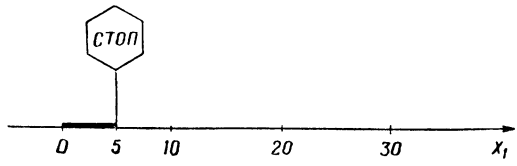
$$5x_1 \leqslant 75,$$
$$6x_1 \leqslant 30,$$
$$x_1 \leqslant 10$$

и, разумеется, $x_1 \geqslant 0$. Нам надо найти наибольшее значение x_1, которое удовлетворяло бы этим трем неравенствам одновременно. Вначале мы определим пространство решений конкретного неравенства, а затем найдем пространство общих решений всех неравенств.

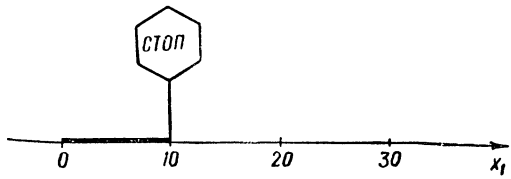
Для $5x_1 \leqslant 75$ мы получим, что любое значение x_1 между 0 и 15 удовлетворяет этому ограничению.



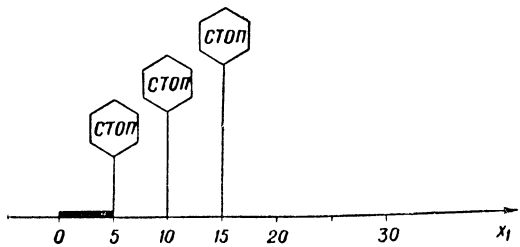
Для $6x_1 \leqslant 30$ пространство решений расположено между 0 и 5.



И наконец, для $x_1 \leqslant 10$ множество решений заключено между 0 и 10.



Если мы изобразим все стоп-сигналы на одном графике, то получим

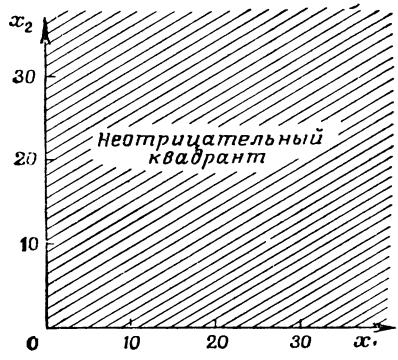


Таким образом, при движении от точки $x_1 = 0$ мы не сможем перейти точку $x_1 = 5$. Любое значение, заключенное между 0 и 5, будет удовлетворять трем неравенствам одновременно; это множество точек и будет множеством решений нашей задачи. Если мы ищем наибольшую точку во множестве решений, то оптимальным ответом исходной задачи с тремя неравенствами будет $x_1 = 5$. Таким образом мы можем найти оптимальное решение для любой задачи с одной переменной.

ДВУМЕРНЫЕ ЗАДАЧИ

Теперь мы вступаем на арену, где развертывается действие двумерных задач, и именно здесь будет нетривиальным и поучительным принять вызов и решить линейные задачи. Чтобы исследовать эту область, я должен сделать некие допущения относительно математической изощренности читателя. Для понимания дальнейшего нужно знать, как изображаются графически двумерные ограничения (уравнения и неравенства), а также иметь представление об основах высшей алгебры. Хотя я и предполагаю такой уровень знаний, но постараюсь оперировать по возможности простыми понятиями, чтобы каждый читатель смог вступить на арену во всеоружии.

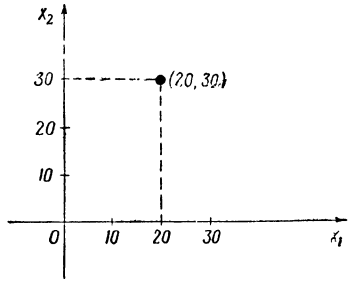
Две переменные (или измерения) x_1 и x_2 мы изображаем с помощью прямоугольных осей.



Поскольку область изменения наших переменных ограничена множеством неотрицательных значений, мы можем двигаться вдоль осей только в определенных направлениях. Так, вдоль оси x_1 мы можем двигаться только вправо от точки 0, начала координат, а вдоль оси x_2 нам разрешается подниматься только вверх от начала координат, перпендикулярно оси x_1. Выражаясь языком математики, пространство решений представляет собой неотрицательный квадрант.

Значение $x_1 = 20$ посылает нас на 20 единиц вправо, а значение $x_2 = 30$ ведет нас на 30 единиц вверх. Взя-

тые вместе, эти направления приводят нас внутрь неотрицательного квадранта, как показано на рисунке.

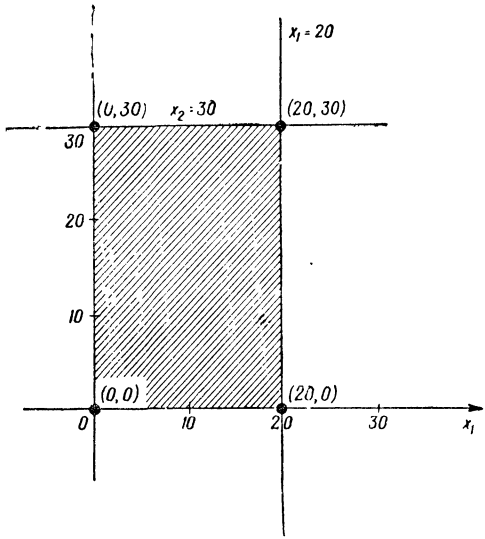


Мы пользуемся сокращенным обозначением точки на плоскости $(x_1, x_2) = (20, 30)$.

В случае одного измерения пространством решений неравенства $x_1 \leqslant 20$ служит область между $x_1 = 0$ и $x_2 = 20$. Любая точка из этой области удовлетворяет требованию, содержащемуся в данном неравенстве: неотрицательная переменная x_1 должна быть меньше или равна 20. Таким образом, это пространство решений содержит бесконечное множество решений. Например, 0, 1, 2, 10, $\sqrt{3}$ и $^3/_4$ — допустимые значения x_1. Однако задача состоит не в том, чтобы найти пространство решений, а в том, чтобы отыскать ту точку в данном пространстве, в которой x_1 принимает минимальное значение. Присоединение целевой функции позволяет нам «остановиться» на некоторой частной точке из бесконечного множества решений. Аналогичный графический анализ проводится и в случае двумерной задачи. Мы должны, вообще говоря, вначале определить бесконечное множество решений, удовлетворяющих ограничениям задачи, а затем выбрать в нем точку, которая минимизирует целевую функцию. Чтобы показать, как это делается, мы решим сейчас несколько довольно простых двумерных задач.

Мы хотим максимизировать x_1 при условии, что $x_1 \leqslant 20$ и $x_2 \leqslant 30$, где $x_1 \geqslant 0$ и $x_2 \geqslant 0$. На двумерном графике мы проводим пограничные прямые $x_1 = 20$ и $x_2 = 30$. Условия неотрицательности $x_1 \geqslant 0$ и $x_2 \geqslant 0$ вместе с пограничными прямыми принуждают нас ограничиться в своих поисках нужной точки заштрихованной областью вместе с границами — это и есть наше пространство решений. Максимальное значение x_1 в нашем

пространстве будет равно 20. Однако мы замечаем странную вещь — существует много точек из пространства решений, в которых $x_1 = 20$. Точки (20, 0) и (20, 30) обе находятся в этом пространстве, и для них обеих $x_1 = 20$. Оптимальное решение оказалось не единственным — довольно обычный случай в линейном программировании,



не доставляющий нам лишних хлопот. Фактически у нас бесконечно много оптимальных решений, поскольку любая точка на отрезке прямой, соединяющей точки (20, 0) и (20, 30), есть решение с $x_1 = 20$.

Важная особенность линейного программирования состоит в том, что в задачах, «ведущих себя хорошо» (вроде той, которую мы обсуждаем), в число оптимальных точек входят вершины, образованные пересечением граничных прямых. В нашей задаче таких вершин четыре: (0, 0), (0, 30), (20, 0), (20, 30), причем последние две будут оптимальными решениями, если мы максимизируем x_1.

Если мы изменим целевую функцию и захотим максимизировать x_2, то оптимальными будут вершины (0, 30) и (20, 30), равно как и любая точка на отрезке, их соединяющем. Если бы мы минимизировали x_1 или x_2, то обнаружили бы, что первая целевая функция дости-

гает минимума в вершинах (0, 0) и (0, 30), в то время как вторая — в вершинах (0, 0) и (20, 0). Разумеется, точки, которые лежат на отрезках, соединяющих эти пары вершин, тоже представляют собой оптимальные решения для соответствующих целевых функций.

Усложним теперь нашу задачу, поменяв целевую функцию. Будем искать те значения x_1 и x_2, которые минимизируют сумму $x_1 + x_2$ и удовлетворяют тем же ограничениям, что и названные выше. Эта целевая функция принимает наименьшее значение в точке (0, 0), то есть минимум $x_1 + x_2 = 0$. Данное оптимальное решение единственно, поскольку в любой другой точке пространства решений по крайней мере одна из переменных положительна.

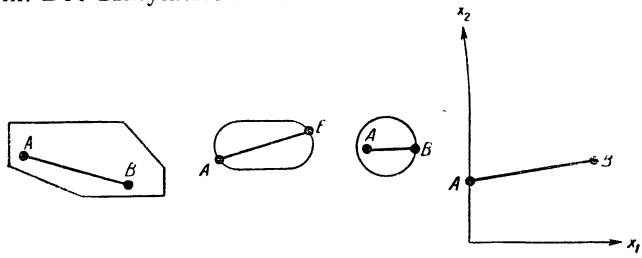
А что можно сказать о решении в случае, если нам надо максимизировать $x_1 + x_2$? Небольшое исследование имеющихся здесь возможностей приводит нас к одному решению (20, 30) с максимумом $x_1 + x_2 = 50$.

Из всего сказанного читатель мог бы заключить, что при любой целевой функции оптимальное решение находится в одной из четырех вершин и, быть может, в некоторых других граничных точках. Это и в самом деле так в любой «хорошо себя ведущей» задаче линейного программирования (задачей, «ведущей себя плохо», будет та, у которой целевая функция не ограничена, как это было в случае нашей первой одномерной задачи). Теперь самое время задать вопрос: ради чего мы, собственно, хлопочем? Но ведь если оптимальное решение задачи линейного программирования находится в одной из вершин, то нам остается только определить все вершины, вычислить в каждой из них целевую функцию и выбрать ту, которая оптимизирует данную функцию. Пусть, например, нам надо максимизировать $3x_1 - 2x_2$. Составим таблицу:

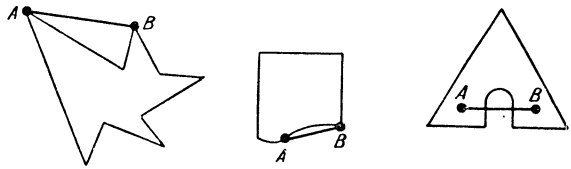
Вершина	Значение $3x_1 - 2x_2$	Максимум
(0,0)	$3x_1 - 2x_2 = 3\,(0) - 2\,(0) = 0$	
(20,0)	$3x_1 - 2x_2 = 3\,(20) - 2\,(0) = 60$	60
(0,30)	$3x_1 - 2x_2 = 3\,(0) - 2\,(30) = -60$	
(20,30)	$3x_1 - 2x_2 = 3\,(20) - 2\,(30) = 0$	

Оптимальное значение равно 60, и единственным оптимальным решением будет точка (20, 0). Читатель может проделать то же самое с любой линейной целевой функцией при данных ограничениях. Все это кажется очень простым и на самом деле таково в подобных «игрушечных» примерах. Но в общем случае трудность при таком подходе состоит в том, что количество вершин может быть огромным, а их перечисление представит собой серьезную вычислительную задачу. Наш подход годится для задач, которые разбираются в этой главе. Однако даже здесь мы можем улучшить его и сделать более похожим на тот алгебраический вычислительный метод, который используется при решении настоящих задач линейного программирования, то есть на *симплекс-метод*. Для этого нам следует взяться за более сложные задачи. Но прежде введем некоторые основные понятия линейного программирования.

Читатели, знакомые с современной математикой, уже должны были догадаться, что пространство решений двумерной задачи представляет собой *выпуклое множество* точек и что вершины суть *крайние точки* этого выпуклого множества. Некоторое множество точек называется выпуклым, если оно вместе с любыми двумя точками содержит и целый отрезок, соединяющий эти точки. Вот выпуклые множества:



А эти множества не являются выпуклыми:



Точка выпуклого множества называется крайней точкой, если она не лежит ни на каком отрезке, соединяю-

щем две другие точки данного множества. Выпуклое множество может содержать конечное или бесконечное число крайних точек или вовсе не содержать таких точек. Предоставим читателю привести примеры множеств каждого типа. Выпуклое множество, связанное с линейной программой, содержит конечное число крайних точек. Вообще говоря, оно представляет собой *выпуклый многогранник.*

ПРОИЗВОДСТВЕННАЯ ЗАДАЧА

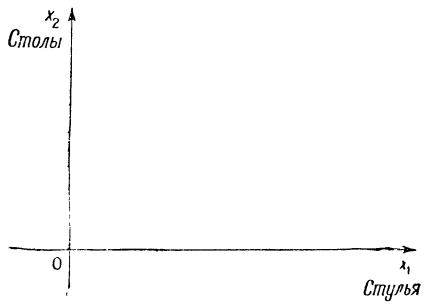
Плох тот план, который не допускает изменений.

Сейчас мы рассмотрим упрощенный вариант производственной задачи, с которой встретилась наша Мебельная Компания Простофиль. Чтобы свести дело к двумерной задаче, мы рассмотрим только производство стульев и столов при наличии ограничений на количество досок и человеко-часов, а также изменим количество наличных ресурсов, чтобы облегчить графическое решение задачи. Таким образом, наша задача состоит в следующем: максимизировать функцию доходов $45x_1 + 80x_2$ (долларов) при условии, что

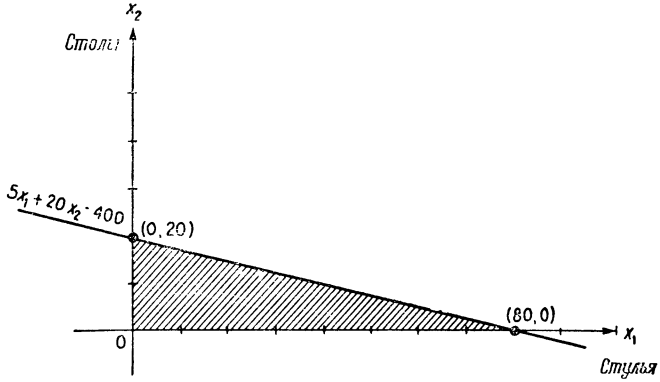
$$5x_1 + 20x_2 \leqslant 400,$$
$$10x_1 + 15x_2 \leqslant 450.$$

Здесь x_1 означает число изготовляемых стульев (c), а x_2 — число столов (t), причем $x_1 \geqslant 0$ и $x_2 \geqslant 0$. Всего у нас имеется 400 футов досок красного дерева и 450 человеко-часов, которые в процессе производства должны превратиться в столы и стулья.

Первое, с чего следует начать, — это определить пространство решений. Но заметим сначала, что оно расположено в неотрицательном квадранте.

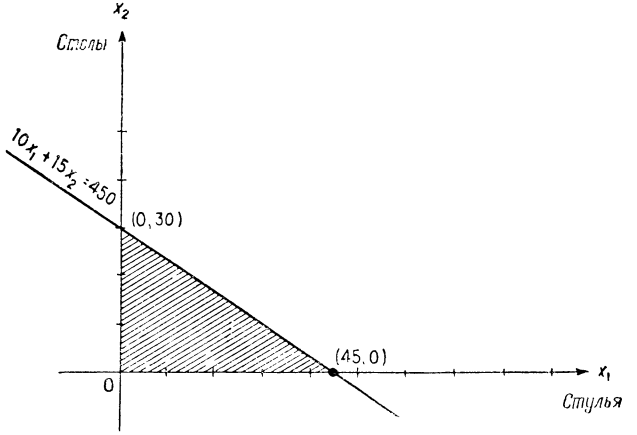


Первое неравенство $5x_1 + 20x_2 \leqslant 400$ требует, чтобы мы оставались в той части неотрицательного квадранта, которая состоит из точек (x_1, x_2) с координатами, удовлетворяющими этому неравенству. Самый легкий способ определить такую область состоит в том, что мы проводим граничную прямую $5x_1 + 20x_2 = 400$. А это мы сможем сделать, отыскав две точки (одну на оси x_1, а другую на оси x_2), через которые проходит данная прямая. Если $x_1 = 0$, то $x_2 = 20$; если $x_2 = 0$, то $x_1 = 80$. Две точки, (80, 0) и (0, 20), лежат на данной граничной прямой.

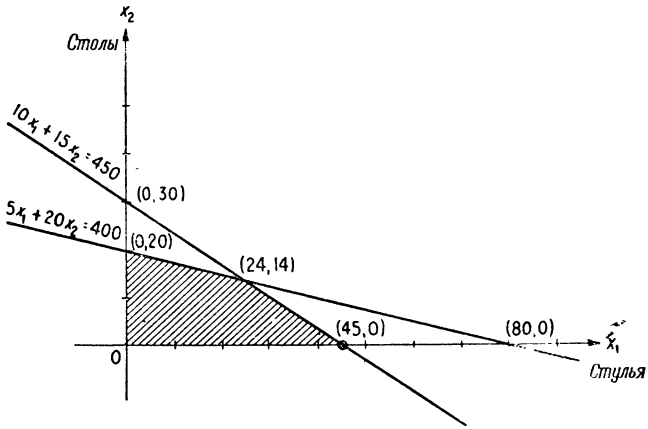


Все точки, лежащие в заштрихованной области и на ее границе, удовлетворяют неравенству $5x_1 + 20x_2 \leqslant 400$.

Для второго неравенства граничной прямой будет $10x_1 + 15x_2 = 450$, а ее точки пересечения с осями координат равны (45, 0) и (0,30), как показано на рисунке.



Любая точка данного выпуклого множества удовлетворяет неравенству $10x_1 + 15x_2 \leqslant 450$. Решение, удовлетворяющее обоим неравенствам, мы найдем, наложив обе заштрихованные области друг на друга на одном графике — их общая часть и есть искомое пространство решений.

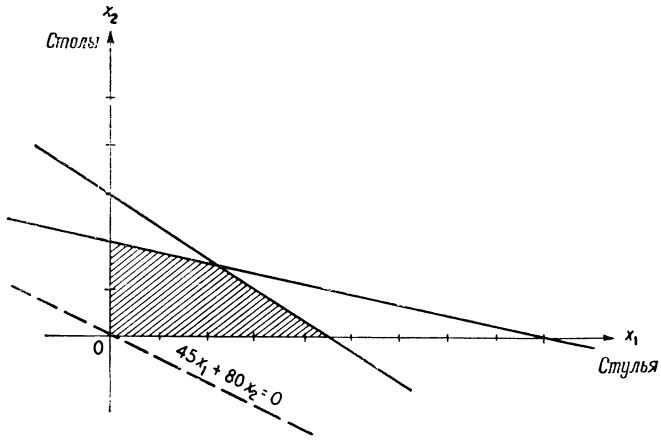


Это объединенное пространство решений представляет собой все возможные комбинации столов и стульев, которые можно изготовить при наличии 400 футов досок и 450 человеко-часов. Только точки, принадлежащие этой области, удовлетворяют ограничениям задачи.

До сих пор мы нашли все крайние точки нашего выпуклого множества решений, за исключением одной, которая получается при пересечении двух пограничных прямых. Эта точка представляет собой тот случай, когда мы полностью расходуем все наши ресурсы на производство столов и стульев. Любая точка прямой $5x_1 + 20x_2 = 400$, например (0, 20), требует, чтобы мы использовали ровно 400 футов досок. Аналогично любая точка прямой $10x_1 + 15x_2 = 450$ требует полного использования всех имеющихся в наличии человеко-часов. С помощью нехитрого алгебраического трюка мы найдем, что точка $x_1 = 24$ и $x_2 = 14$ как раз и лежит на обеих наших прямых.

Задача состоит в том, чтобы определить, какая из бесконечного множества точек заштрихованной области максимизирует $45x_1 + 80x_2$. Мы можем рассматривать это выражение как левую часть некоторого уравнения, связывающего между собой переменные x_1 и x_2.

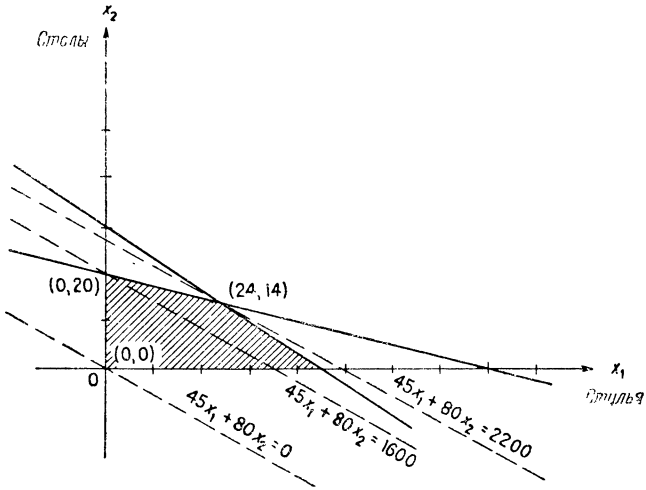
Например, уравнение $45x_1 + 80x_2 = 0$ показано на рисунке пунктирной прямой.



Если Мебельная Компания Простофиль не производит ни столов, ни стульев, то эта прямая показывает, что прибыль равна нулю. Мы хотим, чтобы наша прямая доходов смещалась в направлении возрастания доходов, но так, чтобы удовлетворялись и все ограничения, связанные с наличием ресурсов. Мы можем перемещать нашу прямую доходов по заштрихованной области до тех пор, пока не будем остановлены на границе (мы не можем искать решение вне заштрихованной области). Такое движение прямой доходов показано на рисунке на стр. 89. Оптимальный план производства состоит в том, чтобы изготовить $x_1 = 24$ стульев и $x_2 = 14$ столов, что даст доход $45 \times 24 + 80 \times 14 = 2200$ долларов. «Протаскивание» целевой функции через выпуклое множество как раз и происходит именно тогда, когда мы применяем специальные алгебраические приемы симплекс-метода.

В симплекс-методе мы начинаем с любой крайней точки, вроде $x_1 = 0$, $x_2 = 0$, определяем, что существуют лучшие решения, расположенные в крайних точках, и испытываем одно из них. При этом процессе мы улучшаем решение, переходя от какой-либо крайней точки к лучшей (в нашем случае к $x_1 = 0$, $x_2 = 20$), и продолжаем так действовать до тех пор, пока не дойдем до крайней точки, не допускающей дальнейшего улучшения.

Отличительная черта симплекс-метода, которая делает его столь экономным, состоит в том, что с его помощью мы находим путь от одной крайней точки к лучшей соседней, перебирая только небольшое подмножество множества всех крайних точек. В этом отношении процесс оказался очень эффективным, хотя никто в точности не знает, почему.

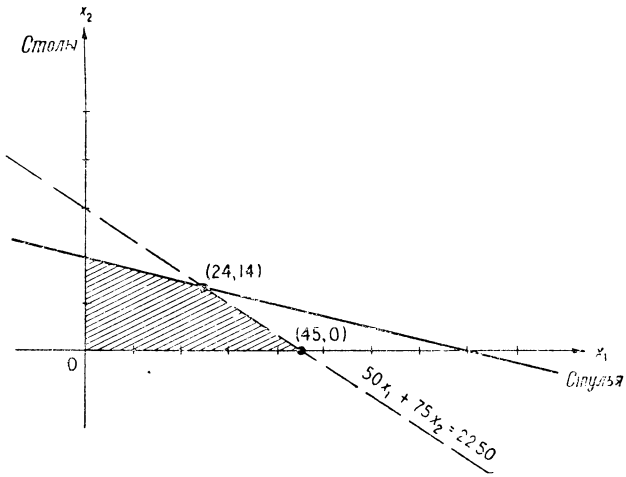


Точка (24, 14) представляет собой оптимальный план производства для широкой области целевых функций. Если доход от продажи одного стула или стола меняется, то меняется и наклон прямой доходов — при этом на рисунке мы наблюдаем поворот прямой доходов вокруг точки (24, 14). Однако достаточно большое изменение коэффициентов дохода так накренит нашу прямую, что оптимум окажется в другой точке. Для любой прямой доходов оптимум достигается в одной из четырех крайних точек. Неоднозначность оптимального решения возникает тогда, когда целевая функция занимает предельное положение, включающее границу выпуклого множества. Для целевой функции $50x_1 + 75x_2$ оптимальными будут крайние точки (24, 14) и (45, 0), так же как и все точки на границе, их соединяющей (см. рис. на стр. 90).

Такой подход можно использовать при решении любой двумерной задачи. Нам придется найти только те точки, которые нужны для проведения пограничных пря-

мых, да еще крайнюю точку, в которой целевая функция последний раз соприкасается с выпуклым множеством решений.

Подобную общую геометрическую интерпретацию можно дать и для многомерных задач. Мы имеем выпуклый многогранник, определенный с помощью данных ограничений, и хотим «пропустить» через него целевую функцию в направлении, ведущем к оптимуму. Читатель



может наглядно представить себе этот процесс, рассматривая комнату, в которой он сидит, как выпуклое множество, определенное с помощью ограничений некоторой линейной программы. Один угол комнаты представляет собой начало координат этого трехмерного мира. Пол, стены и потолок — это границы нашего выпуклого множества, расположенного в неотрицательном октанте трехмерного пространства. Любая целевая функция вроде $x_1 + x_2 + x_3$ принимает оптимальное значение в одной из восьми крайних точек. Если мы отбросим ограничение, задаваемое потолком, то наше выпуклое множество станет неограниченным. Если x_3 — вертикальное направление, а мы захотим максимизировать x_3, то целевая функция x_3 будет неограниченной. Пусть читатель сам представит себе все разнообразие ситуаций, возникающих в трехмерном мире. Вновь подчеркнем, что задачи более чем с двумя переменными не решаются графически,

однако многие из них можно решить с помощью карандаша, бумаги и симплекс-метода.

Читатель уже, вероятно, заметил, что оптимальное решение нашей производственной задачи (24 стула и 14 столов) потребовало всех наличных ресурсов. При этом решении ограничение на количество красного дерева, которое в общем случае имеет вид неравенства $5x_1 + 20x_2 \leqslant 400$, превращается в равенство $5(24) + 20(14) = 400$. Точно так же для человеко-часов получим $10 \times 24 + 15 \times 14 = 450$. Возникает естественный вопрос: что произойдет с доходами нашей компании, если она сможет увеличить количество некоторых ресурсов? Кроме того, если Компания Простофиль при данных ресурсах захочет изготовлять не только столы и стулья, но и письменные столы, то как ей решить, выгодно ли переключать ресурсы на производство письменных столов? Подобные вопросы и ответы на них являются основными для экономической теории и управления производством. Первый вопрос относится к так называемому маргинальному анализу, в то время как второй имеет дело с конъюнктурными издержками. Для той линейной модели, которая встретилась в нашей задаче о столах — стульях (фактически для любой линейной модели), ответ на эти вопросы можно получить, решая вспомогательную, так называемую двойственную задачу[1]. Двойственная задача формулируется по данным исходной задачи. Задача, двойственная к нашей, выглядит следующим образом: минимизировать функцию стоимости ресурсов $400w_1 + 450w_2$ при условии, что

$$\begin{aligned} 5w_1 + 10w_2 &\geqslant 45, \\ 20w_1 + 15w_2 &\geqslant 80, \\ w_1 &\geqslant 0, \\ w_2 &\geqslant 0. \end{aligned}$$

Здесь w_1 — неизвестная стоимость (учетная цена) единицы ресурсов красного дерева, а w_2 — неизвестная стои-

---

[1] Подробнее о двойственных задачах и, в частности, о задаче двойственной к производственной задаче можно прочесть в книге С. Гасса «Линейное программирование (методы и приложения)» (М., Физматгиз, 1961). — *Прим. перев.*

мость единицы ресурсов рабочей силы. (Читателю следует обратить внимание на то, что коэффициенты целевых функций и правые части неравенств в этих задачах меняются местами. Следует также заметить, что коэффициенты, стоявшие по строкам в исходной задаче, расположены по столбцам в двойственной к ней задаче.) Поскольку эти ресурсы представляют собой «узкие места» в оптимальном решении, то есть лимитируют валовой продукт, нам хотелось бы надеяться, что в оптимальном решении двойственной задачи эти двойственные цены будут положительны. Оказывается, что оптимальное решение имеет вид: $w_1 = 1$ и $w_2 = 4$ со значением целевой функции, равным 2200 долларам; мы истолковываем этот факт следующим образом.

Доход от дополнительного вклада одной единицы красного дерева равен 1; одной единицы рабочей силы — 4. Если один из ресурсов, например красное дерево, имеется в избытке, то его дальнейшее увеличение не приведет к возрастанию прибыли и, следовательно, его «дополнительная прибыльность» (маргинальное значение) будет равна нулю. Основная теорема линейного программирования (теорема двойственности) утверждает, что если исходная задача имеет конечное оптимальное решение, то это справедливо и для двойственной к ней задачи, причем оптимальные значения целевых функций совпадают. Так, мы отметим, что минимальная цена всех ресурсов равна валовому доходу в 2200 долларов.

Поскольку в оптимальном решении исходной задачи требуется, чтобы производились как стулья, так и столы, конъюнктурные издержки равны нулю. Это обстоятельство отражается и на оптимальном решении двойственной задачи, поскольку при $w_1 = 1$ и $w_2 = 4$ оба ограничения обращаются в равенства. Если бы, например, в оптимальном решении требовалось изготовлять только одни столы, то конъюнктурные издержки для стульев, равные разности между левой и правой частями первого неравенства $5w_1 + 10w_2 - 45$ были бы положительны, то есть стоимость ресурсов, идущих на производство стула, превышала бы доход от одного стула. Таким образом, затраты на производство стула больше тех, которые получатся, если эти же ресурсы направить на производство столов. Подобный же анализ можно провести, чтобы выяснить, выгодно ли производить третье наименование

продукции, например письменные столы. Читателю следовало бы самому решить графически задачу, двойственную к задаче о столах — стульях, и ответить на вопрос: если доход от продажи письменного стола равен 110 долларам, а на производство такого стола идет 15 единиц красного дерева и 25 единиц рабочей силы, то следует ли Мебельной Компании Простофиль изготовлять эти столы?

Одно из важных достоинств линейного программирования состоит в его способности отвечать на подобные вопросы. На практике очень часто нам гораздо важнее знать конъюнктурные издержки и дополнительную прибыльность, чем само оптимальное решение.

ЗАДАЧА О ДИЕТЕ (ВНОВЬ)

Будь прост в одежде и в еде излишеств бойся;
Короче, поцелуй меня и успокойся.

Для того чтобы продемонстрировать геометрическую ситуацию, слегка отличную от предыдущей, мы вернемся к задаче о приготовлении завтрака, которую решала наша хозяйка. Вместо K мы будем писать x_1, а вместо $C - x_2$. С математической точки зрения задача состоит в том, чтобы найти неотрицательные значения x_1 и x_2, минимизирующие общую стоимость

$$3{,}8x_1 + 4{,}2x_2$$

при условии, что

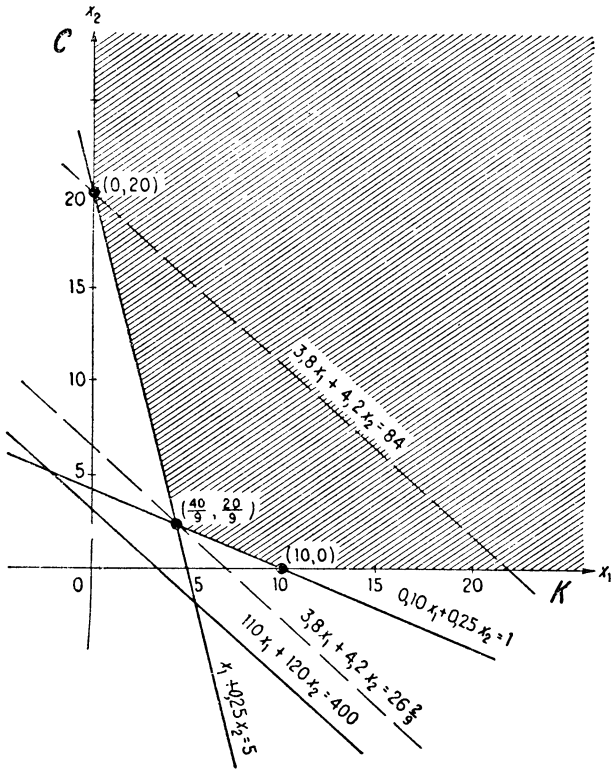
$$0{,}10x_1 + 0{,}25x_2 \geqslant 1,$$

$$1{,}00x_1 + 0{,}25x_2 \geqslant 5,$$

$$110{,}00x_1 + 120{,}00x_2 \geqslant 400.$$

Проведя, как и выше, граничные прямые, мы получим заштрихованное пространство решений, изображенное ниже. Это выпуклое множество не является выпуклым

многогранником и не ограничено ни в направлении x_1, ни в направлении x_2.



Графический анализ показывает, что ограничения, наложенные на калории, будут выполнены, если будут удовлетворены все прочие ограничения. Граничная прямая калорий $110x_1 + 120x_2 = 400$ не является частью границы нашего выпуклого множества решений. Она излишняя в нашей задаче, и поэтому мы можем в дальнейшем ее не рассматривать. На рисунке мы изобразили пунктиром две прямые стоимости: $3{,}8x_1 + 4{,}2x_2 = 84$ (цента) и прямую минимальной стоимости $3{,}8x_1 + 4{,}2x_2 = 26^2/_9$ (цента). Оптимальное решение единственно, в нем требуется, чтобы хозяйка купила $x_1 = {}^{40}/_9$ унций K и $x_2 = {}^{20}/_9$ унций C.

Двойственной к нашей будет следующая задача: найти неотрицательные значения w_1, w_2 и w_3, максимизирующие величину

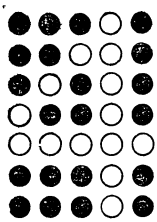
$$w_1 + 5w_2 + 400w_3$$

при условии, что

$$0{,}10w_1 + 1{,}00w_2 + 110{,}00w_3 \leqslant 3{,}8,$$
$$0{,}25w_1 + 0{,}25w_2 + 120{,}00w_3 \leqslant 4{,}2.$$

Здесь w_i можно интерпретировать как стоимость единицы питательного вещества i. Следовательно, в двойственной задаче ищутся такие w_i, при которых цена минимальной диеты принимает свои максимальные значения, причем предполагается, что цена питательных веществ в единице каждого вида не превосходит его стоимости.

Использование геометрии в линейном программировании позволяет нам рассматривать бесконечное множество решений как совокупность точек в пространстве надлежащим образом выбранной размерности. Множество решений выпукло, и в силу этого обстоятельства мы можем, за исключением случая бесконечно большого максимума, ограничиться в своих поисках конечным, хотя и, как правило, довольно большим множеством крайних точек. Очарование и мощь линейного программирования обязаны эффекту взаимодействия плодотворных вычислительных методов и широкого круга важных приложений — одно немыслимо без другого.



ЛИНЕЙНОЕ ПОПУРРИ

(Всякая всячина,
собранная на полях
Страны Линейного Программирования.)

Любые попытки дать полный обзор предметов, задач и приложений, на которые оказывает влияние линейное программирование, обречены на неудачу. Эти области слишком широки и многообразны, чтобы их можно было охватить в одной книге. В предыдущих главах я рассказал о многих наиболее важных, хотя и стандартных, задачах из Страны Линейного Программирования. В этой главе мы продолжим обсуждение, включив сюда как дополнительные приложения, так и некоторые необычные вопросы, связанные с линейным программированием.

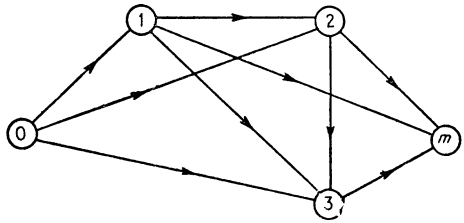
СЕТЕВЫЕ ЗАДАЧИ

Очнись, о мой Сент-Джон! Брось мелкие дела,
Чтоб гордость королей обуздана была.
И коль от жизни мы уж ничего не ждем,
Лишь поглядим окрест и в лучший мир уйдем.
Так подивимся же сему людскому балагану;
Огромен лабиринт, хоть он и не без плана.

Довольно важный класс задач линейного программирования можно объединить под общим названием сетевых задач. Поскольку подход к формулировке математических моделей, с ними связанных, довольно непосредствен, мы проиллюстрируем его на нескольких небольших примерах.

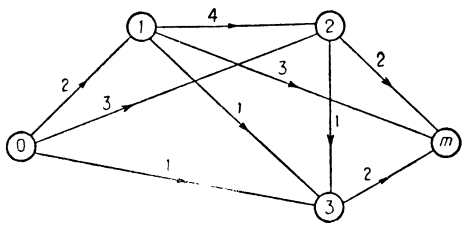
Пусть нам дана транспортная сеть (трубопроводы, железные дороги, телефонная сеть), по которой мы хотим послать однородные единицы (нефть, машины, сообщения) из некоторой точки сети, пункта отправления, в определенное место, называемое пунктом назначения. Кроме этих двух пунктов, сеть состоит из множества промежуточных узловых пунктов, соединенных путями

между собой и с данными двумя пунктами. Эти узловые пункты можно интерпретировать как места, в которых происходит переход с одного пути на другой. Мы будем обозначать пункт отправления через 0, пункт назначения через m, а узловые пункты — цифрами или буквами. Вот типичный пример небольшой сети, который мы обсудим в дальнейшем.



Путь, связывающий узловые пункты *2* и *3*, мы обозначим символом (2, 3) или в общем случае (i, j). Пути, изображенные на нашей сети, представляют собой направленные пути, так как поток грузов вдоль каждого из них происходит в направлении, указанном стрелкой. Если грузы могут передвигаться в двух направлениях, то соответствующие узлы связываются двумя противоположно направленными путями.

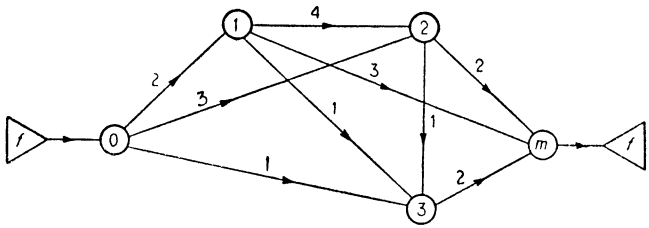
Вдоль каждого пути может передвигаться только конечный поток грузов. Так, участок трубопровода может пропускать только определенное количество нефти в час, линия связи может обслужить только определенное число вызовов в день и т. д. Таким образом, с каждым путем связана верхняя граница того потока, который этот путь может через себя пропустить — его пропускная способность. Мы обозначим, например, через f_{23} максимальный поток, пропускаемый путем (2, 3). На помещенном здесь рисунке мы указываем такие верхние границы для каждого из путей, в данном случае $f_{23} = 1$.



Поток некоторых грузов отправляется из пункта 0 и, двигаясь вдоль путей, попадает в промежуточные узловые пункты, затем движется вдоль других путей в другие узловые пункты или в пункт назначения до тех пор, пока все грузы, которые начали свой путь в пункте 0, не попадут в пункт m. Другими словами, мы накладываем на сеть условие *сохранения потока* в промежуточных узлах: то, что попадает в такой узел, должно и выйти из него. Это предположение аналогично закону Кирхгоффа для электрических цепей.

Сетевая задача о максимальном потоке

Первая задача, которую мы рассмотрим, — сетевая задача о максимальном потоке — состоит в том, чтобы послать как можно большее количество грузов из пункта 0 в пункт m. Мы предполагаем, что в пункте 0 существует неограниченный запас грузов и что единственным препятствием для пересылки этих грузов в пункт m служит пропускная способность путей, ограниченная заданными верхними границами. Таким образом, мы хотим определить максимальный поток (обозначим его через f), который можно послать из пункта отправления в пункт назначения, используя пути с заданной пропускной способностью. Мы хотим пропустить f через нашу сеть так, как показано на рисунке.



Математическую модель для подобной сети построить нетрудно. Определяя

1) переменные x_{ij} как количество грузов, доставленных из узлового пункта i в узловой пункт j, и вспоминая, что

2) f — общее количество грузов, пропущенных через сеть,

3) f_{ij} — верхняя граница потока на пути (i, j),

4) выполнено допущение о сохранении потока,

5) поток f и переменные x_{ij} неотрицательны,

мы получаем следующую задачу линейного программирования:

максимизировать

$$f$$

при условии, что

$$\begin{aligned}
x_{01} + x_{02} + x_{03} &= f,\\
-x_{01} + x_{12} + x_{13} + x_{1m} &= 0,\\
-x_{02} - x_{12} + x_{23} + x_{2m} &= 0,\\
-x_{03} - x_{13} - x_{23} + x_{3m} &= 0,\\
-x_{1m} - x_{2m} - x_{3m} &= -f,\\
x_{01} &\leqslant 2,\\
x_{02} &\leqslant 3,\\
x_{03} &\leqslant 1,\\
x_{12} &\leqslant 4,\\
x_{13} &\leqslant 1,\\
x_{1m} &\leqslant 3,\\
x_{23} &\leqslant 1,\\
x_{2m} &\leqslant 2,\\
x_{3m} &\leqslant 2,
\end{aligned}$$

где $x_{ij} \geqslant 0$ и $f \geqslant 0$.

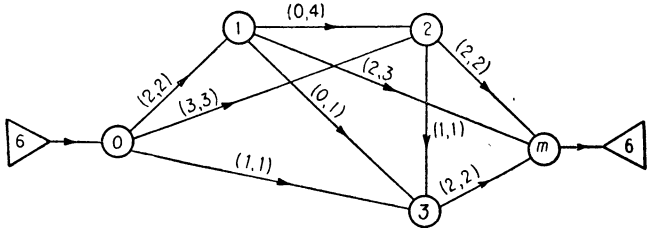
Первое уравнение устанавливает, что поток f, попадающий в пункт 0, равен суммарному потоку, направляющемуся из 0 в другие узловые пункты. Аналогично интерпретируются и остальные уравнения. Для каждого узлового пункта выписано свое уравнение. Неравенства представляют собой ограничения сверху на потоки вдоль каждого пути.

Задача, разумеется, разрешима, поскольку все ограничения удовлетворяются, если $x_{ij} = 0$ и $f = 0$. Однако это, как говорят, тривиальное решение.

Хотя задачу о максимальном потоке можно решить с помощью симплекс-метода, математическая структура подобных задач позволяет применять методы решения, которые быстрее и легче приводят к цели. Даже весьма объемистые задачи с тысячами узловых пунктов и путей оказываются вполне посильными. Однако мы не будем

углубляться в сущность этих методов, хотя и упомянем некоторые факты, к ним относящиеся.

В простом случае нашей сети максимальный поток $f = 6$. На следующем рисунке мы поставили около каждого пути (i, j) по паре чисел, например (0, 4), где первое число, x_{ij}, равно потоку вдоль данного пути, а второе, f_{ij}, — верхней границе потока вдоль данного пути. Разумеется, $x_{ij} \leqslant f_{ij}$. В случае оптимального решения мы имеем



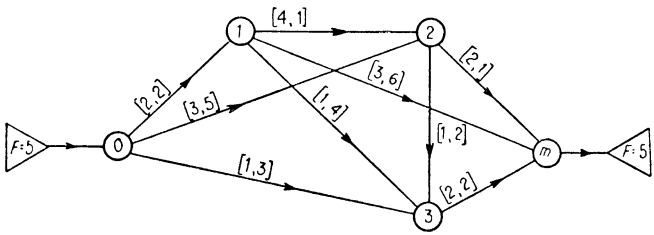
Читатель заметит, что пути, выходящие из пункта 0, насыщены — мы не можем увеличить соответствующие потоки. Если мы удалим эти пути — (0, 1), (0, 2) и (0, 3), то наш пункт отправления окажется отрезанным от пункта назначения. Такое множество путей мы назовем *сечением*. Сечение — это такое множество путей, которое, будучи удаленным из сети, разделяет ее на две не связанные между собой части, в одной из которых находится пункт отправления, а в другой — пункт назначения. Пути, принадлежащие некоторому сечению, имеют определенную общую пропускную способность. В нашем случае она равна 6. Для сечения, состоящего из путей (0, 3), (1, 3), (1, m), (2, 3) и (2, m), общая пропускная способность равна 8. Основная теорема теории сетей гласит: для любой сети максимальный поток из пункта 0 в пункт m равен минимальной пропускной способности среди всех сечений, разделяющих 0 и m. Подумав минуту, каждый скажет, что эта теорема интуитивно очевидна. Алгебраическое доказательство займет немного больше времени. Эффективные вычислительные методы для задачи о максимальном потоке основаны на результатах данной теоремы. Кроме того, эти методы гарантируют, что значения переменных (общий поток и поток вдоль каждого пути) будут выражаться целыми числами.

Сетевая задача о потоке минимальной стоимости

Другой важный класс сетевых задач составляют задачи о потоке минимальной стоимости. Как мы увидим далее, к этому типу задач относятся среди прочих транспортная задача и задача о кратчайшем пути.

В задаче о потоке минимальной стоимости, как и в задаче о максимальном потоке, нам дана некоторая сеть, по которой надо доставить единицы однородного груза из пункта отправления в пункт назначения. С каждым путем (i, j) связана стоимость c_{ij} транспортировки единицы груза из пункта i в пункт j. Мы должны доставить определенное количество F единиц груза из пункта отправления в пункт назначения так, чтобы общая стоимость транспортировки была минимальной. Мы предполагаем, что в узлах выполняется условие сохранения потока и что поток x_{ij} вдоль пути (i, j) неотрицателен и ограничен, то есть $0 \leqslant x_{ij} \leqslant f_{ij}$. Математическая модель этой задачи совершенно аналогична модели задачи о максимальном потоке, и мы проиллюстрируем ее на примере сети из предыдущего раздела. Мы должны помнить, однако, что общий поток F нам теперь известен (в задаче о максимальном потоке общий поток f требовалось найти).

Пусть в нашем примере $F = 5$ (он не может превосходить 6). Стоимость транспортировки c_{ij} единицы груза вдоль некоторого пути и верхнюю границу f_{ij} для данного пути мы укажем на рисунке с помощью пары чисел $[f_{ij}, c_{ij}]$. В нашем случае имеем



Так, например, для пути (2, 3) $f_{23} = 1$ и $c_{23} = 2$.

Математическая модель состоит в том, чтобы минимизировать

$$2x_{01} + 5x_{02} + 3x_{03} + x_{12} + 4x_{13} + 6x_{1m} + 2x_{23} + x_{2m} + 2x_{3m}$$

при условии, что

$$
\begin{aligned}
x_{01} + x_{02} + x_{03} &= 5,\\
-x_{01} + x_{12} + x_{13} + x_{1m} &= 0,\\
-x_{02} - x_{12} + x_{23} + x_{2m} &= 0,\\
-x_{03} - x_{13} - x_{23} + x_{3m} &= 0,\\
-x_{1m} - x_{2m} - x_{3m} &= -5,\\
x_{01} &\leqslant 2,\\
x_{02} &\leqslant 3,\\
x_{03} &\leqslant 1,\\
x_{12} &\leqslant 4,\\
x_{13} &\leqslant 1,\\
x_{1m} &\leqslant 3,\\
x_{23} &\leqslant 1,\\
x_{2m} &\leqslant 2,\\
x_{3m} &\leqslant 2
\end{aligned}
$$

и все $x_{ij} \geqslant 0$.

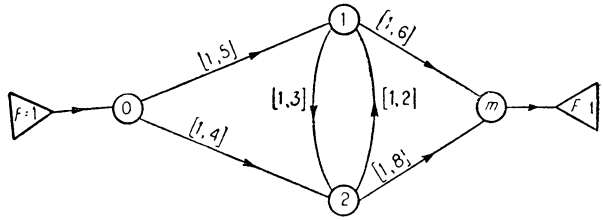
Мы не будем касаться специального метода, предназначенного для решения таких задач, а просто скажем, что с его помощью очень эффективно решаются весьма громоздкие задачи. Однако мы можем разобрать крайне интересный частный случай этой задачи, для которого существует довольно простой метод решения.

Задача о кратчайшем пути

Представим себе, что наша сеть — это карта дорог, пункт отправления — город, из которого мы отправляемся, пункт назначения — город, куда мы едем, а c_{ij} — расстояние между городами (узловыми пунктами). Тогда, полагая $F = 1$ и верхние границы $f_{ij} = 1$, мы сможем из предыдущей задачи получить другую: найти минимальное расстояние между пунктами отправления и назначения. Мы отправляем одну единицу (поток) из пункта отправления в пункт назначения так, чтобы общее расстояние, пройденное этой единицей, было минимальным. Такую задачу, конечно, можно решать обычными методами линейного программирования, однако существуют более эффективные алгоритмы. Один из них мы

продемонстрируем на примере следующего варианта задачи о кратчайшем пути.

Пусть нам дана сеть, состоящая из четырех городов, соединенных путями, длины которых указаны на рисунке и в таблице.



Из	В		
	1	2	m
0	5	4	—
1	—	3	6
2	2	—	8

Вычислительный метод для подобной задачи представляет собой простую комбинаторную схему и состоит из двух основных шагов.

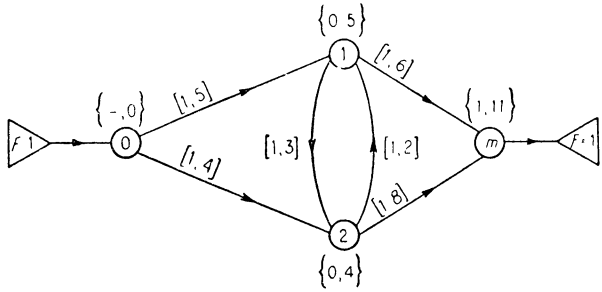
Алгоритм выбора кратчайшего пути

Шаг 1. Дадим каждому узловому пункту по ярлычку вида $\{—, d_i\}$, где первая компонента указывает предыдущий узловой пункт на кратчайшем пути, а d_i обозначает кратчайшее расстояние от пункта 0 до пункта i. Пункту 0 сначала дадим ярлычок с $d_0 = 0$, причем первая компонента этого ярлычка, естественно, и сейчас и в дальнейшем будет равна —. Всем другим узловым пунктам дадим вначале ярлычки с $d_i = \infty$ (любое очень большое число) и с первой компонентой —.

Шаг 2. Возьмем любой путь (i, j), для которого $d_i + c_{ij} < d_j$, заменим ярлычок в узле j на $\{i, d_i + c_{ij}\}$ и продолжим этот процесс до тех пор, пока не останется ни одного такого пути. В первом случае мы определим более короткий путь из 0 в j, проходящий че-

рез i; в последнем случае процесс закончен, и ярлычок в узле j показывает кратчайшее расстояние от пункта 0 до пункта j.

В нашей задаче мы получаем, что кратчайший путь из 0 в m имеет длину 11 и проходит через пункты 0, 1, m. Соответствующие ярлычки показаны на рисунке



Транспортную задачу, о которой рассказывалось в первой главе, легко можно интерпретировать как сетевую задачу о потоке минимальной стоимости. Взяв рисунок, приведенный на стр. 26, читателю следовало бы добавить пункты отправления и назначения таким образом, чтобы почти удовлетворить требованиям задачи о потоке минимальной стоимости. Пункт отправления содержит $F = 25$ холодильников, из которых 11 следует отправить на первую фабрику, а 14 — на вторую. Пункт назначения должен принять 25 холодильников, причем 10 из первого магазина, 8 из второго и 7 из третьего. Эти условия могут стать частью математической модели, если мы добавим к ней ограничения, требующие, чтобы из пункта отправления на соответствующие фабрики было послано 11 и 14 холодильников, а в пункт назначения прибыло ровно 10, 8 и 7 холодильников из соответствующих магазинов. Условия такого типа внесут небольшие изменения в общий алгоритм и могут быть включены в общую задачу о потоке минимальной стоимости.

ЗАДАЧА КОММИВОЯЖЕРА

Боевой клич коммивояжера «Узнай получше свой район!» обычно напоминает нам о том, что коммивояжер должен хорошо знать, где ему искать предполагаемых клиентов, а также какие товары и в каком стиле

предпочитают постоянные и предполагаемые покупатели. Поглядите, вот уже многие годы коммивояжер Мебельной Компании Простофиль Вилли Саймон путешествует по обширной территории, изучая страну и добиваясь внушительных рекордов по продаже товаров. Вместе с тем, однако, весьма внушительны и его расходы. Консультанты по вопросам управления решили при случае заглянуть и на территорию Вилли. Очень скоро Вилли ужасно утомили вопросы приблизительно такого характера: «Вилли, а что произошло 3 июня в Бостоне? Расходы в этот день были слишком велики». Один из этих ребят увязался за Вилли. Во время его путешествия он снимал копии со всех документов и заметок. Вилли попытался подружиться с консультантом, чтобы повлиять на его отчет в благоприятную для себя сторону. Но когда Вилли узнал, что консультант математик, то, порывшись в своем коммивояжерском запасе притч и историй, угостил его таким анекдотом.

Лошадиный разум

Один профессор математики решил провести свой отпуск в деревне. Местный фермер, узнав, что его новый сосед — математик, пришел к нему и рассказал о своей лошади, которая владела четырьмя действиями арифметики. Профессор, дабы посмеяться над фермером, нанес визит его лошади. Действительно, лошадь могла делать все, о чем рассказал фермер. Давая правильный ответ на простые задачи, она либо нужное число раз била о пол конюшни копытом, либо, взяв в зубы мел, писала ответ на доске. Профессор решил заняться этой лошадью и вскоре обучил ее основам алгебры и евклидовой геометрии.

Когда пришло время возвращаться, профессор взял лошадь с собой в университет в качестве студента. В течение первого семестра лошадь без труда освоила тригонометрию. И во втором семестре она получала только пятерки, но лишь до тех пор, пока дело не дошло до аналитической геометрии. Тут-то бедная лошадь и споткнулась. Она просто никак не могла одолеть этот предмет. Не помогли и специально приставленные репетиторы.

Математический факультет, не желая терять своего лучшего студента, созвал экстренное общефакультетское

собрание, дабы решить, что можно предпринять. Были приглашены консультанты — ветеринары, психиатры, жокеи, но и они не смогли дать ответа. Еще раз просмотрели ее зачетную книжку: «отлично» по арифметике, алгебре, евклидовой геометрии, тригонометрии; но аналитическая геометрия, казалось, явилась для лошади подлинным камнем преткновения.

Наконец после долгих раздумий один ассистент блестяще решил дилемму. «Нам давно следовало бы догадаться, — сказал он. — Пытаясь обучить лошадь аналитической геометрии, мы просто ставили Декарта перед лошадью» [1].

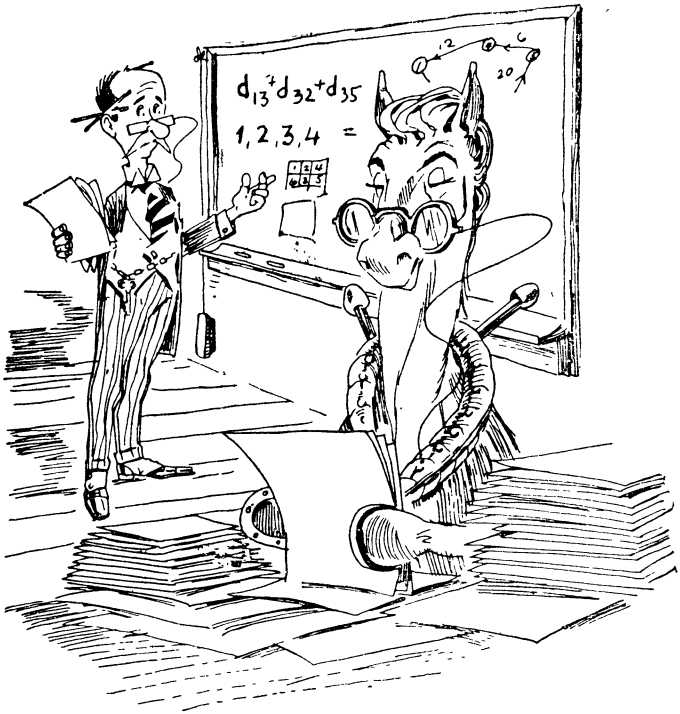
История Вилли не произвела никакого впечатления. Вилли начал выходить из себя, и во время ближайшего визита в контору фирмы выместил все свое раздражение на братце, президенте Саймоне. Вилли пригрозил, что уволится, если его методы и расходы не оставят в покое. Президент Саймон успокоил Вилли, заверив, что консультанты по вопросам управления одобрили все его действия, за исключением одного момента. Исследование показало, что Вилли слишком много времени и денег тратит на путешествия из одного города в другой. Похоже, что он не планирует поездки и не заботится о том, как сделать проделанное им расстояние по возможности короче.

— Единственное, чего мы хотим от тебя, Вилли, — сказал президент Саймон, — это чтобы ты посещал города в том порядке, какой укажут консультанты. Остальные расходы — на твое усмотрение.

Почувствовав большое облегчение, Вилли согласился действовать в соответствии с планом, как только его получит (до сих пор он все еще ждет этот план). Похоже, что смеяться последним придется Вилли, ибо оказалось, что, хотя задачу коммивояжера и можно сформулировать как частный случай задачи линейного программирования, размер территории Вилли (включающей в себя 1000 городов, поселков и деревушек) делает

---

[1] Автор обыгрывает английское выражение to put the cart before the horse — «впрягать телегу перед лошадью», то есть действовать заведомо неправильно. Английское the cart («телега») в американском произношении звучит очень похоже на «Декарт». Декарт (1556—1650) — выдающийся французский философ и математик, основоположник аналитической геометрии. — *Прим. перев.*



ее слишком трудной для того, чтобы провести все вычисления до конца. В чем же состоит трудность?

Будем немного более точными и поставим задачу коммивояжера следующим образом.

Коммивояжер должен посетить каждый город на своей территории. Он выезжает из города, где проживает, посещает каждый город ровно один раз и наконец возвращается домой. Требуется найти маршрут, при котором общее расстояние, проделанное коммивояжером, будет минимальным.

Существует много подходов, позволяющих рассматривать эту задачу как задачу линейного программирования. Однако математически они слишком сложны, чтобы их здесь рассматривать. При данных постановках задачи переменные принимают только целые значения 0 или 1, а число уравнений весьма велико. Например, для задачи Вилли потребовалось бы 10 000 уравнений.

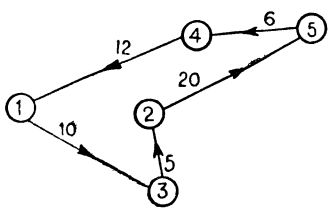
Можно показать некоторые трудности, здесь возникающие, на примере небольшой задачи, включающей в себя только пять городов. Перенумеруем города числами 1, 2, 3, 4 и 5, причем город 1 — это город, в котором живет коммивояжер. Мы знаем расстояния между всеми городами и предполагаем, что из каждого города можно попасть в любой другой. Чтобы сделать рассуждения более общими, предположим также, что расстояние, скажем, от города 2 до города 5 не совпадает с тем расстоянием, которое придется проделать в обратном направлении, из города 5 в город 2. Несимметричная ситуация может возникнуть, например, из-за одностороннего движения, объездов и т. п.

В случае наших пяти городов мы имеем следующую таблицу расстояний между ними в километрах.

Город	1	2	3	4	5
1	0	17	10	15	17
2	18	0	6	12	20
3	12	5	0	14	19
4	12	11	15	0	7
5	16	21	18	6	0

Например, расстояние от города 1 до города 5 $d_{15}=17$, в то время как расстояние от города 5 до города 1 $d_{51}=16$.

Когда коммивояжер покидает город 1, он может поехать в любой из оставшихся пяти городов, пусть для определенности он едет в город 3. Из города 3 он может поехать в любой из двух оставшихся городов (2, 4 и 5) и выбирает 2. Далее он едет в город 5, потом, разумеется, в город 4 и возвращается в город 1.



Такой порядок городов (1, 3, 2, 5, 4, 1) назовем «обходом». Каждому обходу соответствует определен-

ное расстояние. Для нашего обхода общее пройденное расстояние равно

$$d_{13} + d_{32} + d_{25} + d_{54} + d_{41} = 10 + 5 + 20 + 6 + 12 = 53 \text{ (км)}.$$

В задаче о пяти городах существует $4 \times 3 \times 2 \times 1 = 24$ возможных обхода. В подобных небольших задачах мы могли бы перебрать все циклы и соответствующие им расстояния и указать тот цикл, у которого расстояние минимально. Однако даже при незначительном увеличении числа городов такой подход становится непрактичным. Для 10 городов существует 362 800 обходов, а для 15 городов число обходов равно 87 178 291 200.

Подходы к данной задаче, в которых используется линейное программирование, немного напоминают формулировку задачи о назначении персонала. При некотором обходе мы хотим «назначить» каждому городу по пути, связывающему один город с другим, причем так, чтобы одному городу разрешалось иметь только один путь, из него выходящий (человека разрешается назначать только на одну работу, и он обязательно должен быть куда-то назначен). Пути, выходящие из первого города, обозначим через x_{12}, x_{13}, x_{14}, x_{15} и получим уравнение

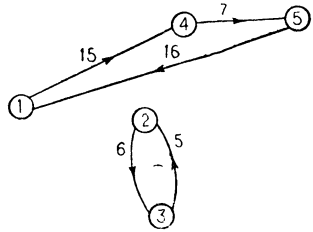
$$x_{12} + x_{13} + x_{14} + x_{15} = 1.$$

Нулевое значение переменной означает, что данный путь не использован; значение, равное единице, говорит нам об обратном — данный путь использован. Для обхода из нашего примера получим следующую таблицу:

Город	1	2	3	4	5
1	0	0	1	0	0
2	0	0	0	0	1
3	0	1	0	0	0
4	1	0	0	0	0
5	0	0	0	1	0

Трудность, возникающая при трактовке задачи коммивояжера как задачи о назначении персонала, состоит в том, что оптимальное значение, при котором общее расстояние минимально, не обязано быть обходом. Общее расстояние 49 километров для следующей таблицы

меньше, чем расстояние 53 километра для нашего обхода, однако, как показано на рисунке, такое решение состоит из двух «подобходов».



Город	1	2	3	4	5
1	0	0	0	1	0
2	0	0	1	0	0
3	0	1	0	0	0
4	0	0	0	0	1
5	1	0	0	0	0

Мы можем добавить условия, которые устранят подобную возможность. Вот почему при подходе, использующем линейное программирование, требуется так много уравнений.

Однако Вилли Саймон не долго будет радоваться. Существуют другие подходы к решению задачи коммивояжера, хотя в самой большой решенной задаче фигурировали только 70 городов. Метод динамического программирования, по-видимому, подходит для небольших задач, в которых число городов не превосходит 13. Задача с 70 городами решалась с помощью специальной схемы, в то время как методы линейного программирования использовались при решении задачи с 42 городами. Задачу Вилли можно разбить на более мелкие задачи, рассматривая близко расположенные города как один город. Однако, как и во всех подобных случаях, аналитики, вооруженные своими алгоритмами и ЭВМ, должны доказать, что решения, полученные ими математически, лучше и их можно пустить в дело.

Мы предоставим читателю определить, будет ли расстояние при нашем обходе (1, 3, 2, 5, 4, 1) минимальным, и если нет, то каков минимальный обход. Понятия

задачи коммивояжера можно распространить и на другие области (почтовые перевозки, оптимальная схема перевозки школьников). Нам не следует бояться таких задач из-за возникающих здесь вычислительных трудностей. Все же здесь можно кое-чего добиться, а порой услышать забавную историю.

ЗАДАЧА О КОНТРАКТАХ

«Продано!»
Возглас аукционера

Где бы мы ни встретились с оптимальной задачей, нам, естественно, захочется выбрать такое допустимое решение, которое в то же время будет и оптимальным. Мы хотим делать все как можно лучше (но поступаем ли мы так на самом деле?). Применяя вычислительные методы линейного программирования, мы берем некоторое решение и ищем лучшее, затем — еще лучшее и т. д. Поэтому естественно задать вопрос: а нужно ли использовать всю эту технику для того, чтобы сэкономить всего-навсего какое-то несчастное пенни? Ведь труд и средства, затраченные на поиски решения лучшего, чем то, которое получено вручную, могут обойтись гораздо дороже достигнутой при этом экономии. Надо ли нам искать диету минимальной стоимости или мы прекрасно обойдемся просто хорошей диетой, удовлетворяющей нашу дневную потребность в питательных веществах? Могли бы мы обойтись решениями Сида Саймона при планировании производства или нам действительно требуется навести экономию?

В некоторых задачах можно обойтись просто хорошим решением, но, вообще говоря, если позволяют средства, лучше все-таки отыскать и использовать оптимальное решение. Однако существуют задачи, в которых совершенно необходимо найти оптимальное решение. Так, в США ведомству заготовок предписано Конгрессом заключать контракты с частными фирмами таким образом, чтобы минимизировать расходы, иными словами, чтобы общие затраты были наименьшими из возможных.

Заключение контрактов может представлять собой довольно запутанную задачу управления, а ограничения, которые накладывают поставщики, могут быть весьма сложными. Поскольку после заключения контрактов все

предложения фирм подлежат публикации, лица, производящие соответствующие расчеты, должны быть весьма проницательны при манипуляциях с этими данными. Однако с помощью линейного программирования можно избавиться от этого тяжкого бремени. Как же справляется с такого рода трудностями линейное программирование? Давайте познакомимся с этой задачей поближе.

Для какого-то вида заготовок некоторая фирма делает свои предложения, в которых указываются:

1) цена единицы товара или товаров;

2) максимальное или минимальное количество единиц каждого наименования, которое фирма может изготовить по установленной цене;

3) любые прочие условия, которые она сочтет нужным выдвинуть.

Эти предложения отражают желание фирмы получить прибыль, ее догадки относительно предложений, сделанных конкурентами, и ее собственные специфические ограничения.

Изучая такие предложения, ведомство заготовок должно прибавить к ценам, установленным каждым поставщиком, некоторые сопутствующие расходы, например расходы на транспорт. Аналогично из этих цен вычитаются все затраты, сэкономленные в силу некоторых специфических условий. Так, например, уменьшение расходов может быть вызвано рассрочкой платежа. По существу задача о контрактах формулируется как транспортная задача, но ввиду некоторых условий, предъявляемых поставщиками, приходится вносить в нее ряд изменений. Например, фирма могла потребовать, чтобы заказ был сделан не менее чем на 500 единиц и не более чем на 1000 единиц. Или могли быть назначены разные цены на каждую партию из 1000 единиц из-за скидки на количество проданного товара. Следующий пример представляет собой настоящую задачу о контрактах, с которой столкнулось ведомство заготовок. Мы только приведем данные, сюда относящиеся, и предоставим читателю самому выписать уравнения и попытаться найти решение.

Перечисленным ниже в таблице складам требуется определенное количество упаковок нужной маркировки.

Местоположение склада	Требуется	
	для внутренних нужд	на экспорт
Колумбус	10 395	
Ричмонд	12 420	
Сан-Антонио	10 395	
Скенектади	9 720	39 555
Юта	3 240	
Шарп	5 535	
Обурн	3 645	
Атланта	10 330	3 510
Итого	65 680	43 065
Всего	108 745	

Это количество делится на две категории: для внутреннего пользования и экспортную, поскольку специальная экспортная упаковка стоит дороже.

Четыре поставщика делают предложения. Поставщику 1 требуется заказ не менее чем на 11 500 упаковок, причем должно быть не более 33 145 упаковок для внутреннего пользования и не более 3510 экспортных. Поставщик 2 не предъявляет никаких требований относительно максимального и минимального числа упаковок. Поставщик 3 предлагает максимум 60 000 упаковок, а поставщик 4 — только упаковки для внутреннего пользования и не выдвигает никаких других условий. Вся эта информация вместе с ценой одной упаковки для каждой комбинации поставщик — склад представлена в таблице:

Город	Требуется	Максимальное предложенное количество				
		Поставщик 1	Поставщик 1	Поставщик 2	Поставщик 3	Поставщик 4
		Для внутрен. нужд	Экспорт			
		33 145	3 510	108 745	60 000	65 680
Колумбус	10 395	0,7289	—	0,6868	0,6574	0,6832
Ричмонд	12 420	0,7398	—	0,7058	0,6489	0,6724
Сан-Антонио	10 395	0,7229	—	0,7204	0,6904	0,7227
Скенектади (для внутр. нужд)	9 720	0,7406	—	0,7075	0,6318	0,6627
Скенектади (на экспорт)	39 555	—	0,7749	0,7319	0,6452	—
Юта	3 240	0,7247	—	0,7358	0,6944	0,7306
Шарп	5 535	0,7276	—	0,7389	0,6973	0,7339
Обурн	3 645	0,7297	—	0,7389	0,6973	0,7339
Атланта (для внутр. нужд)	10 330	0,7325	—	0,7049	0,6646	0,6917
Атланта (на экспорт)	3 510	—	0,7663	0,7291	0,6816	—

Указанные цены включают в себя расходы по доставке. Так, например, для того чтобы купить одну упаковку

у поставщика 2 и доставить ее в склад в Колумбусе, требуется 0,6868 доллара. Решение, удовлетворяющее всем условиям, предъявляемым поставщиками, и минимизирующее общие расходы ведомства заготовок, представлено в следующей таблице:

Город	Стоимость расходов в долларах			
	Поставщик 1	Поставщик 2	Поставщик 3	Поставщик 4
Колумбус	—	—	—	10 395
Ричмонд	—	—	—	12 420
Сан-Антонио	—	10 395	—	—
Скенектади				
на внутр. нужды	—	—	4 515	5 205
на экспорт	—	—	39 555	—
Юта	—	—	3 240	—
Шарп	—	—	5 535	—
Обурн	—	—	3 645	—
Атланта				
на внутр. нужды	—	—	—	10 330
на экспорт	—	—	3 510	—

Общая сумма расходов составляет при этом 72 953,1935 доллара. Методы линейного программирования, с помощью которых было получено данное решение, гарантируют, что лучшего результата добиться невозможно.

ТЕОРИЯ ИГР

Игра — дело серьезное.

Можно считать, что теория игр, как и линейное программирование, относится к современным областям математики. Неискушенному человеку может показаться, что на этом сходство между ними и заканчивается. Ибо в то время, как линейное программирование изучает вопрос о наиболее эффективном распределении ограниченных ресурсов, теория игр занимается разработкой такой стратегии, которая позволит нам добиться наибольшего выигрыша. Однако между этими задачами существует замечательная взаимосвязь, состоящая в том,

что математическая модель некоторого класса задач теории игр совпадает с моделью линейного программирования.

В общем случае теория игр занимается исследованием следующей задачи: пусть дано n игроков (обозначим их $P_1, P_2, \ldots, P_n$), играющих в заданную игру. Как должен играть каждый игрок, чтобы добиться наилучших результатов? В конце партии каждый из игроков получает определенную сумму денег, называемую «выигрышем». Для проигравшего игрока этот «выигрыш» равен потоку монет, «перетекших» из его кармана в карманы более счастливых соперников. Если все деньги, как выигранные, так и проигранные, остаются у игроков, то мы говорим, что эта игра с «нулевой суммой». Примером игры с ненулевой суммой служит покер, при котором игорный дом берет определенный процент с каждой ставки.

Игры классифицируются также по числу игроков и числу возможных ходов. Например, шахматы представляют собой игру с двумя игроками и конечным числом ходов (если мы введем подходящее «правило остановки»), а покер — это игра со многими игроками, но также с конечным числом ходов. Дуэль, при которой оба дуэлянта могут открыть огонь в любой момент данного интервала времени, есть игра с двумя игроками и бесконечным числом возможных ходов. Далее, игра может быть кооперативной и некооперативной. В первом случае игроки могут объединяться и действовать сообща, а во втором каждый играет только сам за себя. Следовательно, игра с двумя игроками, конечно, не может быть кооперативной. Мы обсудим только конечные игры с двумя игроками и нулевой суммой, поскольку именно этот тип игр тесно связан с линейным программированием.

В качестве первого примера рассмотрим теперь уже ставший классическим анализ настоящей стратегической ситуации [1].

Игра с двумя стратегиями

В феврале — марте 1943 года японский конвой судов собрался в Рабауле (остров Новая Британия), чтобы

[1] O. G. Haywood, Jr., Military Decision and Game Theory, *J. Oper. Res. Soc. Am.*, **2**, № 4, 1954.

двигаться затем в Лае (остров Новая Гвинея). Американское командование решило перехватить этот конвой средствами авиации и нанести ему максимальный урон.

У японского командующего был выбор: послать конвой из 16 кораблей либо севернее Новой Британии, либо южнее этого острова. Каждый переход занимал три дня. Выбор перехода и представлял собой возможные стратегии. Метеосводки (известные обеим сторонам) сообщали, что ожидается дождь и плохая видимость на северном пути и ясная погода на южном.

У американского командующего было две возможности. Он мог сконцентрировать большинство своих самолетов-разведчиков либо на одном пути, либо на другом. Бомбардировщики могли нанести удар по конвою на любом пути вскоре после того, как он будет обнаружен.

Задача американского командующего состояла в том, чтобы найти конвой и подвергать его бомбардировке как можно большее число дней; японский командующий хотел свести число бомбардировок до минимума. И один и другой принимали решения независимо один от другого. Как должен был действовать каждый из них? Но прежде чем выбрать нужные стратегии, приведем некоторые дополнительные сведения.

Располагая данными о состоянии погоды, мобильности самолетов и некоторых дополнительных ограничениях, американский командующий мог составить таблицу (2×2) или, как мы будем ее называть, «матрицу выигрышей», в которой указывалось бы, сколько дней будут продолжаться бомбардировки в зависимости от того, куда будут направлены самолеты-разведчики. Анализ проводится примерно так.

1. Стратегия американцев — сконцентрировать самолеты-разведчики на северном пути.

а. Конвой послан северным путем.

Погода мешает разведке, но можно получить два дня бомбардировок из-за правильной концентрации самолетов-разведчиков.

б. Конвой послан южным путем.

Погода ясная, но из-за ограниченного числа самолетов-разведчиков можно надеяться только на два дня бомбардировок.

2. Стратегия американцев — сконцентрировать самолеты-разведчики на южном пути.

а. Конвой послан северным путем.

Плохая видимость и плохое распределение самолетов-разведчиков приводят только к одному дню бомбардировок.

б. Конвой послан южным путем.

Все складывается очень удачно (хорошая погода, много самолетов-разведчиков); конвой можно бомбить три дня.

Располагая все эти данные в виде матрицы выигрышей, получим

		Стратегии японцев (выбор пути)	
		Северный путь	Южный путь
Стратегии американцев (концентрация самолетов-разведчиков)	Северный путь	2 дня	2 дня
	Южный путь	1 день	3 дня

Для того чтобы рассматривать такую военную ситуацию как задачу теории игр, мы должны предположить, что японский командующий располагает теми же данными (он в состоянии выписать ту же матрицу выигрышей и интерпретировать ее так же, как и мы). Игра имеет нулевую сумму, поскольку день бомбардировок, выигранный американцами, проигран японцами, то есть исход, хороший для одного командующего, будет плохим для другого. Трудности теории игр отчасти вызваны тем, что нужно добиться соглашения между противниками относительно действительной значимости чисел, составляющих матрицу выигрышей. Обсуждение этого вопроса увело бы нас слишком в сторону, поэтому предоставим его трактатам по теории игр.

При анализе нашей простой игры с двумя стратегиями проявляются многие характерные черты теории игр. Приведем нашу матрицу выигрышей к стандартным обозначениям теории матриц и будем представлять себе

игру как матрицу (2×2), у которой строки соответствуют стратегиям максимизирующего игрока (в нашем случае — американского командующего), а столбцы — стратегиям минимизирующего игрока (японского командующего). Наша матрица выигрышей имеет вид

$$\begin{pmatrix} 2 & 2 \\ 1 & 3 \end{pmatrix}$$

Присоединимся теперь к нашим командующим, которые сидят в своих штабах, смотрят на цифры и пытаются определить, как следует поступить. Японский командующий, прекрасный игрок в го, уверен, что изберет наилучшую стратегию. Матрица выигрышей причиняет ему некоторое беспокойство: что бы он ни сделал, конвой, по-видимому, будет обнаружен и подвергнут бомбардировке. Это именно та ситуация, в которой невозможно выиграть. Самое лучшее, на что он может надеяться, — один день бомбардировок. Сумеет ли он добиться такого исхода?

Сравнивая столбцы (выигрыш в случае северной стратегии с выигрышем в случае южной стратегии), японский командующий замечает, что ему не следует выбирать южный путь. Если бы он его выбрал, то подвергся бомбардировкам в течение двух или даже трех дней, в то время как для северного пути соответствующее число дней равно двум или одному. Японский командующий выбирает такую стратегию на основании своей матрицы. Он мог бы послать эти данные прямо американскому командующему, так как последний, будучи стратегом, выделил тот же самый столбец из своей матрицы.

Американец свел теперь квадратную матрицу к матрице, которая состоит из двух строк, соответствующих его стратегиям, и одного столбца, означающего, что японцы должны выбрать северный путь. Новая матрица выигрышей имеет вид

$$\begin{pmatrix} 2 \\ 1 \end{pmatrix}$$

Теперь задача стала совсем простой. Поскольку американский командующий хочет максимизировать число

бомбардировок, он выбирает северный путь. И словно это и в самом деле было рассчитано математиками, он действительно послал самолеты-разведчики на северный путь.

Решение нашей задачи обладает одной особенностью (мы скажем, что задача решена с использованием чистой стратегии), а именно: каждый из командующих с необходимостью выбрал северный путь. Командующему, или игроку, даже не приходится рассматривать другие стратегии (чтобы принять решение, ему не нужно подбрасывать монетку). Причина заключается в том, что матрица нашей задачи содержит так называемую «седловую точку». Седловой точкой некоторой матрицы называется такой элемент этой матрицы, который одновременно минимален в своей строке и максимален в своем столбце (два дня бомбардировок на пересечении стратегий северного пути как раз и будут таким числом). Соответствующие строки и столбец представляют собой оптимальные стратегии обоих игроков, а выигрыш совпадает со значением седловой точки. Причина такого положения дел коренится в некотором консерватизме, присущем теории игр. Поясним нашу мысль подробнее.

Предположим, что некоторая игра задается следующей (3×3) матрицей:

$$\begin{pmatrix} 3 & 5 & 6 \\ 2 & -1 & 3 \\ 0 & 7 & 4 \end{pmatrix}$$

Напомним, что числа в матрице выигрышей записаны по отношению к максимизирующему игроку (например, американскому командующему); поэтому положительное число означает, что выигрывает тот игрок, чьи стратегии пишутся по строкам, отрицательное число означает, что он проигрывает эту сумму, а ноль — что деньги вовсе не переходят из рук в руки. Игрок, чьи стратегии пишутся по строкам (первый игрок, как мы его назовем), смотрит на цифры, стоящие в первой строке, и замечает, что если он выберет первую стратегию, то самое худшее произойдет с ним в том случае, если и второй игрок выберет свою первую стратегию. Тогда первый игрок получит три единицы вместо пяти или шести. Пер-

вый игрок точно так же анализирует две другие строки и находит, что он может выиграть соответственно —1 единицу (то есть проиграть 1 единицу) и 0 единиц. Мы выпишем эти минимальные числа рядом с матрицей

$$\begin{pmatrix} 3 & 5 & 6 \\ 2 & -1 & 3 \\ 0 & 7 & 4 \end{pmatrix} \begin{matrix} ③ \\ -1 \\ 0 \end{matrix}$$

Эти дополнительные числа представляют самое худшее, что может произойти с первым игроком в каждом случае. Первый игрок может выбрать самую лучшую из данных наихудших возможностей и всегда придерживаться стратегии 1 (в самом худшем случае он выиграет по крайней мере 3 единицы). Может ли второй игрок что-либо сделать для того, чтобы уменьшить свой проигрыш?

Проводя аналогичный анализ с точки зрения второго игрока, мы замечаем, что если он выберет первый столбец, то самое худшее для него произойдет, если первый игрок выберет первую строку, то есть когда второй игрок заплатит первому общую сумму в 3 единицы. Для второго и третьего столбцов «выигрыш» будет равен соответственно 7 и 6. Добавляя эти числа к матрице, получим

$$\begin{pmatrix} 3 & 5 & 6 \\ 2 & -1 & 3 \\ 0 & 7 & 4 \end{pmatrix} \begin{matrix} ③ \\ -1 \\ 0 \end{matrix}$$
$$\begin{matrix} ③ & 7 & 6 \end{matrix}$$

Самое лучшее, что в данной ситуации может сделать второй игрок (не считая отказа от игры),— это все время придерживаться стратегии первого столбца и постоянно проигрывать по 3 единицы. Таким образом, элемент 3 нашей матрицы представляет собой седловую точку. В играх с седловыми точками решение находится с помощью анализа, аналогичного проделанному выше. Мы видим, что если игрок отклоняется от своей чистой стратегии, соответствующей расположению (строка или

столбец) седловой точки, то он рискует больше потерять или меньше выиграть.

Однако интересны те игры, у которых нет седловых точек. Если мы попытаемся применить наш анализ к игре, определенной с помощью (3×4) матрицы

$$\begin{pmatrix} 1 & 5 & 0 & 4 \\ 2 & 1 & 3 & 3 \\ 4 & 2 & -1 & 0 \end{pmatrix},$$

то увидим, что наилучшая из наихудших возможностей (так называемый «максимин») для первого игрока реализуется в случае второй строки, а наихудшая из наилучших возможностей («минимакс») для второго игрока реализуется в случае третьего столбца:

$$\begin{array}{cccc} 1 & 5 & 0 & 4 \\ 2 & 1 & 3 & 3 \\ 4 & 2 & -1 & 0 \\ 4 & 5 & \textcircled{3} & 4 \end{array} \quad \begin{array}{c} 0 \\ \textcircled{1} \\ -1 \end{array}$$

В данном случае нет седловой точки. Тем не менее первый игрок может гарантировать выигрыш по меньшей мере в 1 единицу, всегда придерживаясь своей чистой стратегии 2, в то время как второй игрок может гарантировать себе проигрыш не более чем в 3 единицы, придерживаясь своей стратегии 3. В такого рода играх участники могут получить выигрыш выше среднего, если они будут случайным образом выбирать одну из возможных стратегий. В таком случае первый игрок сможет выиграть больше 1 единицы, а второй игрок — потерять меньше 3 единиц. Позволяя себе «смешивать» стратегии и выбирая случайным образом одну из них от партии к партии, игроки могут свести ожидаемую величину выигрыша к числу, заключенному между значением 1 максимина и значением 3 минимакса, то есть каждый из них выгадает по сравнению с чистыми стратегиями. Мы проиллюстрируем понятие смешанной стратегии на примере хорошо известной игры в «чет-нечет».

„Чет — нечет"

В эту игру играют двое. Первый игрок пытается угадать, четна или нет сумма денег в руке у второго («чет» или «нечет»). В нашем варианте этой игры, если первый игрок угадал, то он получает от второго игрока одну единицу (+1), а если не угадал, то платит ему одну единицу (—1). Выигрыш —1 означает, что первый игрок отдает второму одну единицу. Все это для наглядности можно изобразить в виде матрицы

		Выборы второго игрока	
		«чет»	«нечет»
Выборы первого игрока	«чет»	1	—1
	«нечет»	—1	1

Мы видим, что матрица не содержит седловой точки («максимин» равен —1, а «минимакс» +1).

Сейчас мы определим смешанные стратегии для каждого игрока. Первому игроку хотелось бы менять свой выбор «чета» и «нечета» случайным образом, поскольку, если бы он придерживался постоянного правила (например, всегда выбирать «чет»), его противник, тоже игрок неглупый (одно из предположений теории игр), не преминул бы этим обстоятельством воспользоваться. Пусть x_1 — вероятность того, что первый игрок выберет «чет», а x_2 — вероятность того, что он выберет «нечет». Аналогично определим y_1 и y_2 для второго игрока. Типичным будет, например, случай, когда первый игрок выбирает первую стратегию с вероятностью $^3/_4$, а вторую — с вероятностью $^1/_4$. Здесь вероятность интерпретируется в терминах частоты, с которой выбирается та или иная стратегия. В случае смешанной стратегии с $x_1 = {}^3/_4$ и $x_2 = {}^1/_4$ первый игрок, сыграв очень большое число партий, обнаружил бы, что примерно три четверти всего времени пользовался первой стратегией и одну четверть — второй. Поскольку сумма вероятностей, относящихся к каждому игроку, должна равняться единице,

мы имеем $x_1 + x_2 = 1$ и $y_1 + y_2 = 1$, где x и y неотрицательны.

По определению решением игры с нулевой суммой называется пара смешанных оптимальных стратегий (по одной на каждого игрока) и такое число v (цена игры), при котором если первый игрок будет придерживаться своей смешанной оптимальной стратегии, то при любой стратегии второго игрока он выиграет не меньше чем v; а если второй игрок будет придерживаться своей оптимальной смешанной стратегии, то при любой стратегии первого игрока он проиграет не больше чем v.

Сейчас мы перейдем к математической модели нашей задачи, не пытаясь, однако, преувеличивать ее значения. Читателю придется примириться с этим или пропустить несколько страниц.

Первый игрок хочет максимизировать цену игры v при условии, что вероятности $x_1 + x_2 = 1$ и что если он будет придерживаться стратегии (x_1, x_2) против любой чистой стратегии второго игрока, то выиграет по меньшей мере v; другими словами,

$$x_1 - x_2 \geqslant v,$$
$$-x_1 + x_2 \geqslant v.$$

Чтобы получить данные неравенства, мы умножили неизвестные неотрицательные вероятности на соответствующие элементы столбцов матрицы выигрышей, как это показано для первого столбца

$$\begin{matrix} x_1 \times \\ + \\ x_2 \times \end{matrix} \begin{pmatrix} 1 & -1 \\ \cdot & \\ -1 & 1 \end{pmatrix}$$

Каждая такая сумма произведений представляет собой математическое ожидание выигрыша первого игрока при условии, что второй игрок выбирает соответствующую чистую стратегию. Первый игрок хочет, чтобы математическое ожидание его выигрыша было по крайней мере не меньше неизвестной цены игры v. Знак v может быть любым. Положительное v означает, что шансы на выигрыш у первого игрока выше, чем у второго; $v = 0$ означает, что оба игрока имеют равные шансы на выигрыш; отрицательное v означает, что второй игрок имеет более

высокие шансы на выигрыш, чем первый. Первый игрок заинтересован в том, чтобы максимизировать v и найти неотрицательные значения x_1 и x_2, такие, что

$$x_1 - x_2 \geqslant v,$$
$$-x_1 + x_2 \geqslant v,$$
$$x_1 + x_2 = 1.$$

Число v представляет собой тоже переменную нашей задачи, и за исключением того, что v может принимать как положительные, так и отрицательные значения, сформулированная задача относится по существу к линейному программированию. Данную небольшую аномалию можно исправить несколькими способами. Мы можем, например, прибавить большое положительное число w к каждому элементу матрицы выигрышей так, чтобы все элементы стали положительными. Оптимальные стратегии останутся прежними, однако новая цена игры $v + w$ станет положительным числом.

Аналогичную задачу линейного программирования можно поставить и для второго игрока. Она будет состоять в том, чтобы минимизировать v при условии, что

$$y_1 - y_2 \leqslant v,$$
$$-y_1 + y_2 \leqslant v,$$
$$y_1 + y_2 = 1$$

и $y_1 \geqslant 0$, $y_2 \geqslant 0$. Величина v будет прежней. Для игры в «чет-нечет» $v = 0$, то есть это честная игра — игра с равными шансами на выигрыш для обоих игроков.

Задачи линейного программирования для первого и второго игроков тесно связаны между собой в терминах исходной и двойственной задач линейного программирования. (Наши игровые задачи нуждаются лишь в небольшом уточнении, чтобы стать настоящими двойственными задачами.)

Вспоминая геометрический подход к решению задач линейного программирования, рассмотренный в III главе, мы можем легко решать те задачи теории игр, в которых у каждого противника существуют только две стратегии, то есть игры (2×2). [Способ, который мы сейчас опишем, фактически можно распространить на случай $(2 \times m)$ игр, когда у одного игрока только две стратегии, а у другого их m.] Мы решим задачу только для первого игрока и предоставим читателю решить ее для второго. Итак, рассмотрим следующую задачу.

Максимизировать

$$v$$

при условии, что

$$\begin{aligned} x_1 - x_2 &\geqslant v, \\ -x_1 + x_2 &\geqslant v, \\ x_1 + x_2 &= 1, \\ x_1 &\geqslant 0, \\ x_2 &\geqslant 0. \end{aligned}$$

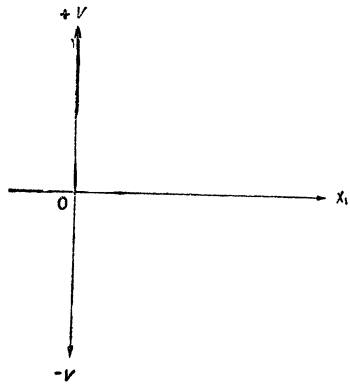
Поскольку геометрический подход к решению таких задач годится только для двух переменных, мы должны свести нашу задачу, в которой участвуют три переменные (x_1, x_2, v) к задаче с двумя переменными. Это можно легко сделать, воспользовавшись уравнением $x_1+x_2 = 1$, с помощью которого одна переменная выражается через другую. Мы полагаем $x_2 = 1 - x_1$ и подставляем это выражение для x_2 через x_1 в наши неравенства, получая новые ограничения:

$$\begin{aligned} x_1 - (1 - x_1) &\geqslant v, \\ -x_1 + (1 - x_1) &\geqslant v, \\ 0 \leqslant x_1 &\leqslant 1, \end{aligned}$$

или

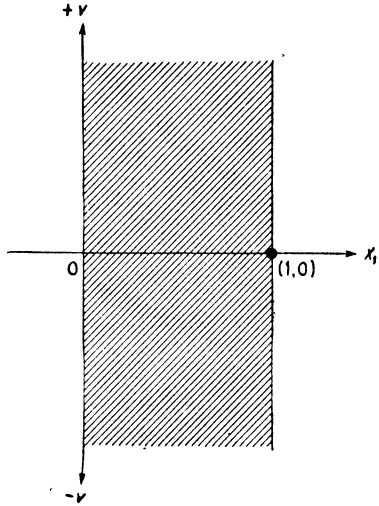
$$\begin{aligned} 2x_1 - v &\geqslant 1, \\ -2x_1 - v &\geqslant -1, \\ 0 \leqslant x_1 &\leqslant 1. \end{aligned}$$

Так как это модель игры первого игрока, мы хотим найти максимальное значение v, помня о том, что v (цена игры) может иметь произвольный знак. Примем x_1 за одну ось координат, а v — за вторую.

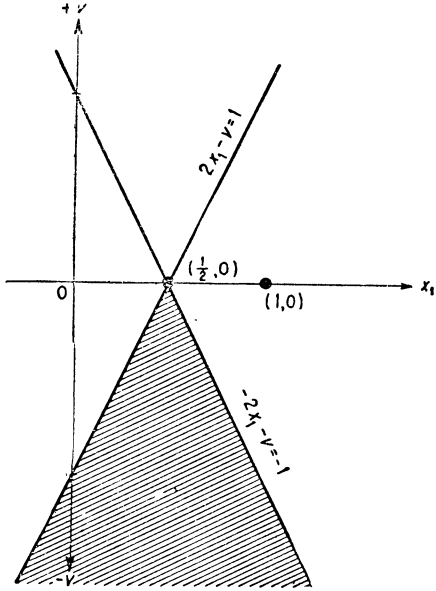


Неравенство $0 \leqslant x_1 \leqslant 1$ требует, чтобы значение x_1 было неотрицательным числом, не превосходящим еди-

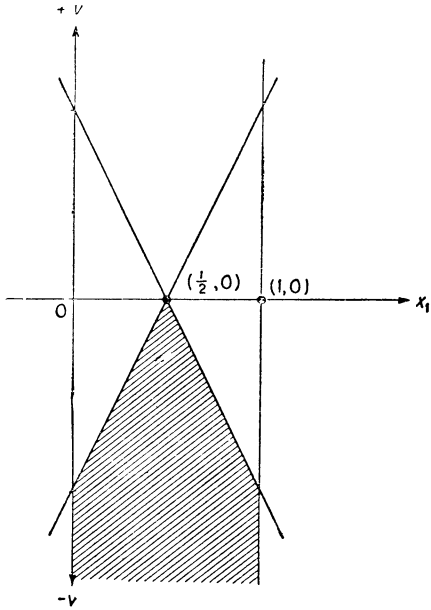
ницы; на рисунке это представлено неограниченной заштрихованной областью.



Проведем теперь прямые $2x_1 - v = 1$ и $-2x_1 - v = -1$ для того, чтобы получить пространство решений обоих неравенств. Это пространство представляет собой заштрихованную область на следующем рисунке:



Накладывая эти два чертежа друг на друга, мы находим точки, удовлетворяющие одновременно трем ограничениям нашей задачи.



Поскольку мы ищем в данной заштрихованной области точку с максимальным v, то оптимальным решением будет точка $x_1 = 1/2$, $v = 0$. Данное решение единственно, поскольку для всех остальных решений значение v отрицательно. Поскольку $x_2 = 1 - x_1$, смешанная оптимальная стратегия для первого игрока состоит в том, чтобы выбирать две свои стратегии случайным образом, с частотой $1/2$ каждую: $x_1 = 1/2$, $x_2 = 1/2$. Эта стратегия гарантирует ему средний выигрыш, равный нулю, то есть игра честная. У второго игрока — та же самая оптимальная стратегия.

Игра „обдираловка"

В поисках нашего последнего примера мы присоединимся к братьям Саймон на ежегодном пикнике Компании Простофиль, приуроченном к сельской ярмарке. Среди гостей находится, конечно, и группа вездесущих

консультантов по вопросам управления. Один из братцев, Си Саймон, попробовав свои силы в бинго, рулетке и других азартных играх, только что подошел к ним. Собственно он вернулся только затем, чтобы призанять немного денег. На обратном пути, уже с пустыми карманами, он набрел на новую игру, в которую, как ему показалось, выиграть можно довольно легко. Игра была карточная, и, познакомившись с ее правилами, Си почувствовал, что нашел оптимальную стратегию. Однако ни один из братьев, знавших, что Си просто рожден для всякого рода проигрышей (он был вице-президентом по вопросам возврата бракованной продукции), не дал ему денег.

Отчаявшись, он подошел к группе консультантов и объяснил, что ему нужно.

— Расскажите-ка нам об этой новой игре, — потребовали они, — и если наш анализ подтвердит ваши слова, то мы уж, так и быть, «скинемся».

— Я точно знаю, что смогу выиграть, — ответил Си, — а правила таковы. Я играю против «зазывалы». У каждого из нас по три карты. У него — бубновый и трефовый тузы и бубновая двойка. У меня тоже есть бубновый и трефовый тузы, но третья карта — двойка треф. Каждый из нас откладывает по одной карте, и затем мы одновременно показываем отложенные карты друг другу. Если масти их разные, выигрываю я, если одинаковые — он. Если отложенными оказываются две двойки, никто не выигрывает. В противном случае сумма выигрыша равна числу очков той карты, которую показал выигравший [1]. Вот и все. Очень просто, не правда ли? Мне нужно узнать только одно — как распорядиться моими трефами, ну, может быть, придется несколько раз воспользоваться бубновым тузом. Я уверен, что выиграю. О, они не зря назвали эту игру «обдираловкой».

Консультанты тут же сообразили, что это игра с двумя игроками, нулевой суммой и тремя возможными стратегиями для каждого партнера. Они сгрудились вместе, записывая что-то на бумажных салфетках, и в конце концов пришли к единодушному решению, что Си следовало бы поискать денег где-нибудь в другом месте,

---

[1] Предполагается, что в данном случае туз эквивалентен одному очку. — *Прим. перев.*

а еще лучше не играть в эту игру вовсе. Их анализ был приблизительно следующим.

Зазывала — первый, максимизирующий, игрок, а Си — второй, минимизирующий, игрок. Матрица выигрышей имеет вид

Стратегия Си

	♢	♧	2♧
♢	1	−1	−2
♧	−1	1	1
2♢	2	−1	0

Зазывала никогда не следует первой стратегии, поскольку он всегда получает столько же или больше, если показывает бубновую двойку; иными словами, выигрыш при третьей стретегии больше или равен соответствующему выигрышу при первой стратегии. Следовательно, на самом деле он играет в приведенную игру (2×3).

$$\begin{array}{c} \\ \clubsuit \\ 2\diamondsuit \end{array} \begin{array}{c} \begin{array}{ccc} \diamondsuit & \clubsuit & 2\clubsuit \end{array} \\ \begin{pmatrix} -1 & 1 & 1 \\ 2 & -1 & 0 \end{pmatrix} \end{array}$$

В этой игре Си никогда не должен придерживаться третьей стратегии (двойка треф), поскольку такой же или еще лучший результат он получит, придерживаясь второй стретегии. Все, наконец, свелось к игре (2×2).

$$\begin{array}{c} \\ \clubsuit \\ 2\diamondsuit \end{array} \begin{array}{c} \begin{array}{cc} \diamondsuit & \clubsuit \end{array} \\ \begin{pmatrix} -1 & 1 \\ 2 & -1 \end{pmatrix} \end{array}$$

Пусть x_2 и x_3 — вероятности того, что зазывала выбирает вторую и третью стратегии (здесь $x_1 = 0$). Модель для этой игры такова. Найти максимальное v при условии, что

$$\begin{aligned} -x_2 + 2x_3 &\geqslant v, \\ x_2 - x_3 &\geqslant v, \\ x_2 + x_3 &= 1, \\ x_2 &\geqslant 0, \\ x_3 &\geqslant 0. \end{aligned}$$

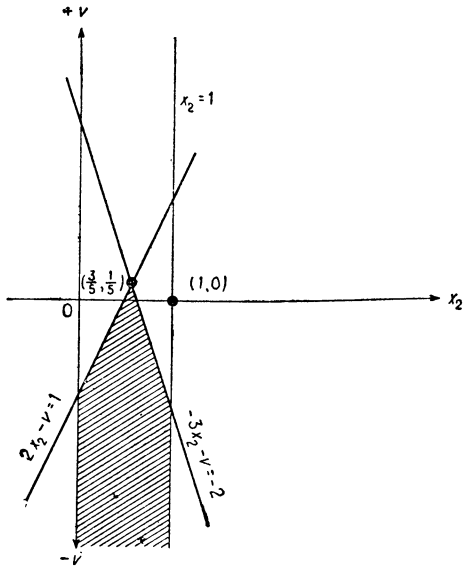
Для того чтобы перейти к двум переменным, мы положим $x_3 = 1 - x_2$ и получим ограничения вида

$$\begin{aligned} -x_2 + 2(1 - x_2) &\geqslant v, \\ x_2 - (1 - x_2) &\geqslant v, \\ 0 \leqslant x_2 &\leqslant 1, \end{aligned}$$

или

$$\begin{aligned} -3x_2 - v &\geqslant -2, \\ 2x_2 - v &\geqslant 1, \\ 0 \leqslant x_2 &\leqslant 1. \end{aligned}$$

Пространство решений представляет собой заштрихованную область в плоскости x_2, v.

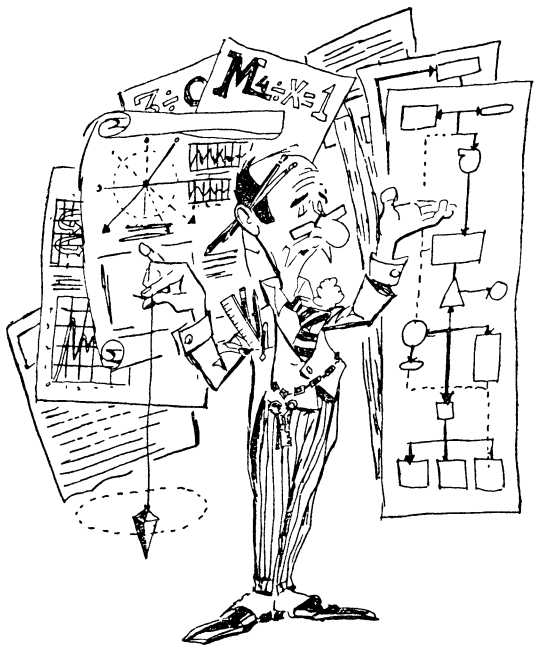


Оптимальной точкой будет $x_2 = {}^3/_5$, $v = {}^1/_5$. Таким образом, цена игры равна ${}^1/_5$. Значит, игра благоприятна для зазывалы, если он никогда не придерживается первой стратегии, вторую стратегию выбирает с вероятностью ${}^3/_5$, а третью — с вероятностью ${}^2/_5$, то есть $x_1=0$, $x_2 = {}^3/_5$, $x_3 = {}^2/_5$ и $v = {}^1/_5$. Эта игра нечестная; зазывала имеет преимущество, и Си пришлось бы еще раз подтвердить свою репутацию. Однако такой логический анализ не произвел на Си ни малейшего впечатления, ибо «уж если повезет, то повезет...»

МАТЕМАТИЧЕСКОЕ ДОПОЛНЕНИЕ И ПЕРЕЧЕНЬ ПРИЛОЖЕНИЙ ЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ

(Что бы ты ни обрел,
помни о конце,
и ты никогда не поступишь неправедно.)

Настоящее дополнение должно помочь читателю построить мост между тем уровнем, на котором велось изложение до сих пор, и уровнем специальных работ по линейному программированию. Чтобы безопасно пройти



по этому мосту, следует заплатить пошлину — ту пошлину, которая чеканится из материалов, разбросанных по самым разным областям математики, методов вычислений, экономики, науки об управлении, инженерного дела и исследования операций. Однако, чтобы выплавить

металл, который поднимет шлагбаум, преграждающий путь на мост, к этим материалам следует еще добавить интуицию и творческие способности.

I. ВВЕДЕНИЕ[1]

В задачах программирования ведутся поиски эффективного распределения заданных ограниченных ресурсов, при котором достигаются желаемые цели. Такие задачи характеризуются большим числом решений, удовлетворяющих основным условиям каждой задачи. Выбор некоторого частного решения в качестве наилучшего решения задачи зависит от конкретной цели, которая указывается при постановке задачи. Решение, удовлетворяющее как условиям задачи, так и заданной цели, называется *оптимальным решением.* Типичным примером подобной задачи служит задача, с которой сталкивается предприниматель, когда хочет распределить наличные ресурсы таким образом, чтобы не только удовлетворить требованиям производства, но и получить максимальную прибыль. Основными условиями данной задачи будут ограничения, наложенные на количество наличных ресурсов и требования производства, а ее целью — желание предпринимателя максимизировать свой доход.

Мы рассмотрим только весьма специальный подкласс задач программирования, называемых задачами линейного программирования. Задачи линейного программирования отличаются от всех прочих тем, что в математических моделях подобных задач используются так называемые «линейные» соотношения. Математически эти соотношения записываются в виде

$$a_1x_1 + a_2x_2 + \ldots + a_jx_j + \ldots + a_nx_n = a_0,$$

где a_j — известные коэффициенты, а x_j — неизвестные переменные. Математическая постановка задач линейного программирования включает в себя систему линейных уравнений или неравенств, выражающих условия

[1] В основу приведенного материала легли работы Гарвина [20], Гасса [21], Хэдли [23] и Вайды [35]. Цифры в скобках означают номер, под которым соответствующие работы приведены в списке литературы.

задачи, и линейную функцию, которая выражает цель задачи.

В настоящем дополнении излагаются основные понятия линейного программирования, дается обзор стандартных вычислительных приемов и перечень весьма разнообразных приложений. Поскольку ограниченный объем не позволяет нам изложить здесь теоретические и математические основы линейного программирования и поскольку вполне очевидно, что нельзя в полной мере воспользоваться мощью этого раздела математики, не изучив его основ, читателю следует познакомиться со специальными работами, приведенными в библиографии.

II. ОПРЕДЕЛЕНИЯ И ОСНОВНЫЕ ТЕОРЕМЫ ЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ

Задача линейного программирования состоит в следующем.

Найти множество чисел $x_1, x_2, \ldots, x_n$, которые минимизируют (или максимизируют) *линейную целевую функцию*

$$c_1x_1 + c_2x_2 + \ldots + c_jx_j + \ldots + c_nx_n \qquad (1)$$

при условии, что выполняются *линейные ограничения*

$$\begin{array}{l} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1j}x_j + \ldots + a_{1n}x_n = a_{10} \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{ij}x_j + \ldots + a_{in}x_n = a_{i0} \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mj}x_j + \ldots + a_{mn}x_n = a_{m0} \end{array} \qquad (2)$$

и *условия неотрицательности*

$$\begin{array}{l} x_1 \geqslant 0 \\ x_2 \geqslant 0 \\ \cdots\cdots \\ x_j \geqslant 0 \\ \cdots\cdots \\ x_n \geqslant 0 \end{array} \qquad (3)$$

Иными словами, можно сказать, что в задаче линейного программирования требуется найти неотрицательное решение системы линейных ограничений, которое оптимизирует, то есть минимизирует или максимизирует, линейную целевую функцию. В этом случае c_j называют коэффициентами стоимости, а a_{i0} — свободными членами. Линейные ограничения могут представлять собой уравнения, уже приведенные выше, или линейные неравенства вида

$$a_{i1}x_1 + \ldots + a_{in}x_n \leqslant a_{i0} \tag{4}$$

или

$$a_{i1}x_1 + \ldots + a_{in}x_n \geqslant a_{i0}. \tag{5}$$

Неравенства вида (4) и (5) можно превратить в уравнения, добавляя или вычитая подходящую *свободную переменную*. В случае (4) мы получили бы, например,

$$a_{i1}x_1 + \ldots + a_{in}x_n + x_{n+1} = a_{i0},\ x_{n+1} \geqslant 0, \tag{6}$$

а в случае (5) —

$$a_{i1}x_1 + \ldots + a_{in}x_n - x_{n+1} = a_{i0},\ x_{n+1} \geqslant 0. \tag{7}$$

Отметим, что у каждого неравенства — своя свободная переменная, с ним связанная. Каждая свободная переменная измеряется разностью между левой и правой частями данного неравенства.

Поскольку в задаче линейного программирования требуется, чтобы все переменные были неотрицательными, мы заметим, что переменные произвольного знака всегда можно выразить как разность двух неотрицательных переменных, например $x_i = x_i' - x_i''$, где $x_i' \geqslant 0$ и $x_i'' \geqslant 0$.

Задача линейного программирования может содержать в себе произвольную смесь линейных ограничений. В вычислительных целях основные ограничения (2) задачи линейного программирования всегда следует задавать в виде уравнений, число которых (m) меньше числа переменных (n). В этом случае система (2) представляет собой *неопределенную* систему линейных уравнений, которая имеет много возможных решений. Поскольку каждое такое уравнение можно рассматривать как уравнение *гиперплоскости* в n-мерном пространстве, *пространство решений* нашей линейной системы представляет собой, вообще говоря, *выпуклый многогранник*. Вы-

числительные методы линейного программирования позволяют среди всех возможных решений определить то, которое оптимизирует целевую функцию. Поскольку пространство решений может быть неограниченным, оптимальное значение целевой функции может также оказаться бесконечно большим. Мы разберем только случай минимизации, так как максимизировать целевую функцию — то же самое, что минимизировать эту же функцию, но взятую с противоположным знаком.

В матричных обозначениях задача состоит в том, чтобы минимизировать

$$\mathbf{cX}$$

при условии, что

$$\mathbf{AX} = \mathbf{b},$$
$$\mathbf{X} \geqslant \mathbf{0},$$

где $\mathbf{c} = (c_1, c_2, \ldots, c_n)$ — вектор-строка; $\mathbf{X} = (x_1, x_2, \ldots, x_n)$ — вектор-столбец; $\mathbf{A}$ — $(m \times n)$-матрица коэффициентов; $\mathbf{b} = (a_{10}, a_{20}, \ldots, a_{m0})$ — вектор-столбец; а $\mathbf{0} = (0, 0, \ldots, 0)$ — вектор-столбец с n строками. Столбцы $\mathbf{A}$ удобно рассматривать как точки в m-мерном пространстве; поэтому задачу линейного программирования можно сформулировать следующим образом: минимизировать

$$\mathbf{cX}$$

при условии, что

$$x_1\mathbf{P}_1 + x_2\mathbf{P}_2 + \ldots + x_n\mathbf{P}_n = \mathbf{P}_0, \quad x_j \geqslant 0,$$

где $\mathbf{P}_j = (a_{1j}, a_{2j}, \ldots, a_{mj})$ — вектор-столбец, а $j = 0, 1, \ldots, n$.

Допустимым решением задачи линейного программирования называется вектор $\mathbf{X} = (x_1, x_2, \ldots, x_n)$, удовлетворяющий условиям (2) и (3).

Основным решением системы (2) называется решение, которое получится, если $n - m$ переменных положить равными нулю и найти оставшиеся m переменных при условии, что определитель, составленный из коэффициентов, стоящих при данных m переменных, отличен от нуля. Эти m переменных называются «основными переменными».

Основным допустимым решением называется основное решение, удовлетворяющее ограничениям (3), то

есть обладающее неотрицательными основными переменными.

Невырожденным основным допустимым решением называется основное допустимое решение, у которого имеется ровно m положительных x_i.

Минимальным допустимым решением называется допустимое решение, минимизирующее (1).

Базисом называется множество линейно независимых векторов. *Допустимым базисом* для задачи линейного программирования называется квадратная матрица $\mathbf{B}$, составленная из множества линейно независимых векторов, выбранных из прямоугольной матрицы $\mathbf{A} = (\mathbf{P}_1, \mathbf{P}_2, \ldots, \mathbf{P}_n)$ так, чтобы для системы $\mathbf{BX}_0 = \mathbf{P}_0$ выполнялось $\mathbf{X}_0 = \mathbf{B}^{-1}\mathbf{P}_0 \geqslant \mathbf{0}$. Например: если $\mathbf{B} = (\mathbf{P}_1, \mathbf{P}_2, \ldots, \mathbf{P}_m)$, то мы получим $\mathbf{X}_0 = (x_{10}, x_{20}, \ldots, x_{m0}) \geqslant \mathbf{0}$. Здесь решением данной задачи будет $\mathbf{X} = (x_{10}, x_{20}, \ldots, x_{m0}, 0, \ldots, 0)$, где последние $n - m$ координат равны нулю. Найти допустимый базис — это все равно, что выбрать определенную «квадратную» систему уравнений из неопределенной «прямоугольной» системы и положить переменные, не входящие в «квадратную» систему, равными нулю.

Выпуклой комбинацией векторов $\mathbf{U}_1, \mathbf{U}_2, \ldots, \mathbf{U}_n$ называется вектор

$$\mathbf{U} = \alpha_1\mathbf{U}_1 + \alpha_2\mathbf{U}_2 + \ldots + \alpha_n\mathbf{U}_n,$$

где α_i — скаляры, $\alpha_i \geqslant 0$ и $\sum \alpha_i = 1$.

Подмножество точек $\mathbf{C}$ евклидова пространства является *выпуклым множеством* в том и только в том случае, если для любой пары точек $\mathbf{U}_1$ и $\mathbf{U}_2$, принадлежащих $\mathbf{C}$, любая их выпуклая комбинация

$$\mathbf{U} = \alpha_1\mathbf{U}_1 + \alpha_2\mathbf{U}_2 = \alpha\mathbf{U}_1 + (1 - \alpha)\mathbf{U}_2, \quad 1 \geqslant \alpha \geqslant 0,$$

также принадлежит $\mathbf{C}$. Выпуклое множество — это такое множество, которое вместе с каждой парой точек содержит и целый отрезок прямой, соединяющий эти точки.

Точка $\mathbf{U}$ выпуклого множества $\mathbf{C}$ называется *крайней точкой*, если $\mathbf{U}$ нельзя выразить в виде выпуклой комбинации двух различных точек из $\mathbf{C}$.

Теорема 1. Множество всех допустимых решений задачи линейного программирования представляет собой выпуклое множество.

Теорема 2. Целевая функция (1) достигает своего минимума в одной из крайних точек выпуклого множества **C**, образованного допустимыми решениями задачи линейного программирования. Если она достигает своего минимума более чем в одной крайней точке, то она достигает его и в любой выпуклой комбинации этих точек.

Теорема 3. Если векторы $\mathbf{P}_1, \mathbf{P}_2, \ldots, \mathbf{P}_k$, где $k \leqslant m$, линейно независимы и таковы, что

$$x_1\mathbf{P}_1 + x_2\mathbf{P}_2 + \ldots + x_k\mathbf{P}_k = \mathbf{P}_0,$$

а все $x_i \geqslant 0$, то точка $\mathbf{X} = (x_1, x_2, \ldots x_k, 0, \ldots, 0)$ есть крайняя точка выпуклого множества допустимых решений. Здесь **X** представляет собой n-мерный вектор, у которого $n - k$ последних координат равны нулю.

Теорема 4. Если $\mathbf{X} = (x_1, x_2, \ldots, x_n)$ — крайняя точка **C**, то векторы, соответствующие положительным x_i, линейно независимы. Отсюда следует, что таких x_i не больше m.

Теорема 5. $\mathbf{X} = (x_1, x_2, \ldots, x_n)$ есть крайняя точка **C** в том и только в том случае, если положительные x_j представляют собой коэффициенты при линейно независимых векторах $\mathbf{P}_j$ в выражении

$$\sum_{j=1}^{n} x_j\mathbf{P}_j = \mathbf{P}_0.$$

Теорема 6. Если существует допустимое решение, то существует и основное допустимое решение.

Теорема 7. Если целевая функция имеет конечный минимум, то по крайней мере одно оптимальное решение является основным допустимым решением.

Эти теоремы позволяют нам ограничиться в наших поисках оптимального решения крайними точками выпуклого множества **C** допустимых решений. Геометрический смысл введенных выше понятий обсуждается в разделе III.

Если мы поставим исходную задачу линейного программирования: минимизировать

$$\mathbf{cX}$$

при условии, что

$$\mathbf{AX} \geqslant \mathbf{b},$$

$$\mathbf{X} \geqslant \mathbf{0},$$

то соответствующая двойственная задача сформулируется следующим образом:
максимизировать

$$\mathbf{Wb}$$

при условии, что

$$\mathbf{WA} \leqslant \mathbf{c},$$
$$\mathbf{W} \geqslant \mathbf{0},$$

где $\mathbf{W} = (w_1, w_2, \ldots, w_n)$ — вектор-строка неизвестных двойственной задачи.

Теорема двойственности. Если одна из задач (исходная или двойственная) имеет конечное оптимальное решение, то другая задача тоже имеет конечное оптимальное решение, а экстремальные значения соответствующих линейных функций совпадают, то есть $\min \mathbf{cX} = \max \mathbf{Wb}$. Если одна из этих задач имеет бесконечное оптимальное решение, то другая вовсе не имеет допустимых решений.

Понятие двойственности и теорема двойственности играют важную роль в теоретическом и вычислительном аспектах линейного программирования.

III. МЕТОДЫ ЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ[1]

1. Симплекс-метод

Основным вычислительным методом, применяющимся для решения *любой* задачи линейного программирования, является *симплекс-метод.* С его помощью мы можем, определив первое основное допустимое решение (крайнюю точку), отыскать минимальное основное допустимое решение за конечное число шагов. Эти шаги, или *итерации*, состоят в том, что мы находим новое основное допустимое решение, для которого соответствующее значение целевой функции будет меньше (или в худшем случае равно) значения целевой функции для предыдущего решения. Данный процесс продолжается до тех пор, пока мы не отыщем минимальное решение

[1] Более подробное изложение материала, приведенного в разделах 1 и 2, читатель может найти в работах, указанных в списке литературы, не претендующем на полноту. Для овладения материалом остальных разделов необходимо обратиться к дополнительной литературе.

с конечным или бесконечным значением целевой функции. Сейчас мы перейдем к математическому описанию *стандартного симплекс-метода.* Слово «симплекс» в названии метода появилось в связи с тем, что в одном из первых примеров, решенных с его помощью, участвовало неравенство $x_1 + x_2 + \ldots + x_n \leqslant 1$, которое определяет симплекс (обобщенный тетраэдр) в n-мерном пространстве, отсекающий на осях координат единичные отрезки.

Допустим, что все координаты вектора $\mathbf{P}_0$ неотрицательны. Мы всегда можем умножить уравнение на -1, чтобы сделать соответствующее $a_{i0} \geqslant 0$. Пусть $\mathbf{B}_1$ — допустимый базис, то есть некоторое основное допустимое решение находится из системы $\mathbf{B}_1\mathbf{X}_{01} = \mathbf{P}_0$, где $\mathbf{X}_{01} = \mathbf{B}_1^{-1}\mathbf{P}_0 \geqslant \mathbf{0}$. На практике $\mathbf{B}_1$ бывает обычно единичной матрицей порядка m, а соответствующее первое допустимое решение находится мгновенно, так как матрица, обратная единичной, снова равна единичной матрице. В этом случае $\mathbf{X}_{01} = \mathbf{P}_0$. В тех же ситуациях, когда в задаче нет подходящей матрицы, можно вначале присоединить к задаче *искусственный единичный базис.* Этот способ обсуждается ниже.

Предположим, что единичный базис для первого допустимого решения существует, и перенумеруем векторы базиса так, чтобы $\mathbf{B}_1 = (\mathbf{P}_1, \mathbf{P}_2, \ldots, \mathbf{P}_m)$ (такой шаг не является необходимым, мы его делаем исключительно для удобства понимания). При этом симплекс-таблица (вычислительная таблица) примет вид

Базис	$\mathbf{P}_0$	$\mathbf{P}_1 \ldots \mathbf{P}_l \ldots \mathbf{P}_m$	$\mathbf{P}_{m+1} \ldots \mathbf{P}_j \ldots \mathbf{P}_k \ldots \mathbf{P}_n$
$\mathbf{P}_1$	x_{10}	$1 \ldots 0 \ldots 0$	$x_{1,m+1} \ldots x_{1j} \ldots x_{1k} \ldots x_{1n}$
.	.	. . .	. . . .
.	.	. . .	. . . .
.	.	. . .	. . . .
$\mathbf{P}_l$	x_{l0}	$0 \ldots 1 \ldots 0$	$x_{l,m+1} \ldots x_{lj} \ldots x_{lk} \ldots x_{ln}$
.	.	. . .	. . . .
.	.	. . .	. . . .
.	.	. . .	. . . .
$\mathbf{P}_m$	x_{m0}	$0 \ldots 0 \ldots 1$	$x_{1,m+1} \ldots x_{mj} \ldots x_{mk} \ldots x_{mn}$
	x_{00}	$0 \ldots 0 \ldots 0$	$x_{0,m+1} \ldots x_{0j} \ldots x_{0k} \ldots x_{0n}$

где $x_{ij} = a_{ij}$ для $i = 1, \ldots, m$ и $j = 0, 1, \ldots, n$; основным допустимым решением будет $\mathbf{X}_{01} = (x_{10}, \ldots, x_{i0}, \ldots, x_{m0}) = \mathbf{B}_1^{-1}\mathbf{P}_0$; и вообще мы можем определить $\mathbf{X}_{j1} = (x_{1j}, x_{2j}, \ldots, x_{mj}) = \mathbf{B}_1^{-1}\mathbf{P}_j$. Значение целевой функции равно $x_{00} = \sum_{i \text{ из базиса}} c_i x_{i0}$. Числа x_{0j} для $j = 1, \ldots, n$ определяются с помощью равенств $x_{0j} = \sum_{i \text{ из базиса}} c_i x_{ij} - c_j$. Суммы, участвующие в этих равенствах, называются косвенными стоимостями и обозначаются иногда через $z_j = \sum_{i \text{ из базиса}} c_i x_{ij}$. Заметим, что $x_{0j} = 0$ для любого j из базиса. Следующие теоремы показывают, зачем нужны x_{0j}.

Теорема 1. Если для любого j выполнено условие $x_{0j} > 0$, то можно построить множество допустимых решений так, чтобы $x'_{00} < x_{00}$ для любого элемента данного множества, где нижняя грань x'_{00} конечна или бесконечна (x'_{00} есть значение целевой функции, принимаемое на некотором элементе множества допустимых решений).

Случай I. Если нижняя грань конечна, то можно построить новое допустимое решение, состоящее ровно из m положительных переменных, на котором целевая функция принимает значение меньшее, чем на предыдущем решении, то есть $-\infty < x'_{00} < x_{00}$.

Случай II. Если нижняя грань бесконечна, то можно построить новое допустимое решение, состоящее ровно из $m + 1$ положительных переменных, на котором целевая функция принимает произвольно малое значение [1].

Теорема 2. Если для произвольного основного допустимого решения $\mathbf{X} = (x_{10}, x_{20}, \ldots, x_{m0})$ условия $x_{0j} \leqslant 0$ выполняются при всех $j = 1, 2, \ldots, n$, то данное решение является минимальным допустимым решением.

В теореме 1 предполагается невырожденность; иными словами, все основные допустимые решения задачи предполагаются строго положительными (все $x_{i0} > 0$). Такое предположение необходимо с теоретической точки зрения, поскольку оно позволяет доказать, что симплекс-метод сойдется за конечное число шагов. Если же

[1] Имеется в виду значение, малое не по абсолютной, а по алгебраической величине. — *Прим. перев.*

нам встретится задача с вырожденным основным допустимым решением, то существует возможность, что процесс *зациклится*, то есть мы будем вновь и вновь возвращаться к исходному базису за конечное число шагов и, следовательно, никогда не получим оптимального решения. Хотя примеры, где встречаются циклы, и построены, они весьма искусственны и обычно процесс сходится как в вырожденной, так и в невырожденной задаче. Кроме того, существуют специальные вычислительные приемы, гарантирующие сходимость в любом случае, хотя обычно ими и не пользуются.

Чтобы определить новое основное допустимое решение, нужно предпринять следующие шаги. С их помощью мы один за другим меняем векторы базиса, пока не доберемся до того места, где следует остановиться.

1. Вычисляем все x_{0j}.

2. Проверяем, все ли $x_{0j} \leqslant 0$ для $j = 1, 2, \ldots, n$. (Это множество неравенств называется критерием оптимальности.) Если все, то текущее решение оптимально и процесс прекращается. Если существует $x_{0j} = 0$ такое, что соответствующий вектор $\mathbf{P}_j$ не входит в оптимальный базис, то мы получим другое оптимальное решение, введя этот вектор в базис. Если нет, введем в базис вектор $\mathbf{P}_k$, для которого $x_{0k} = \max x_{0j} > 0$. Если таких векторов несколько, выбираем любой из них.

3. Чтобы новое решение было допустимым, из базиса следует удалить вектор $\mathbf{P}_l$, для которого

$$\frac{x_{l0}}{x_{lk}} = \min_{x_{ik}>0} \frac{x_{i0}}{x_{ik}}.$$

Если таких векторов несколько, выбираем любой из них. Если все $x_{ik} \leqslant 0$, то задача имеет бесконечное оптимальное решение и процесс останавливается. Если отношение $x_{l0}/x_{lk} \geqslant 0$ окажется равным нулю (вырожденный случай с $x_{l0} = 0$), то значение целевой функции для нового решения будет то же, что и для прежнего. Элемент x_{lk} называется *опорным элементом*.

4. Определяем новое решение и новую симплекс-таблицу по следующей формуле (метод Гаусса):

$$x'_{ij} = x_{ij} - \frac{x_{lj}x_{ik}}{x_{lk}}, \quad i \neq l;$$

$$x'_{lk} = \frac{x_{lj}}{x_{lk}}.$$

Эти формулы справедливы для $i = 0, 1, \ldots, m$ и $j = 0, 1, \ldots, n$. x'_{i1} при $j = 0$ есть новое допустимое решение; x'_{00} — новое значение целевой функции; x'_{0j} равны новым разностям между косвенными и прямыми стоимостями. Такое преобразование эквивалентно определению нового допустимого базиса $\mathbf{B}_2$, для которого новым решением служит вектор

$$\mathbf{X}_{02} = \mathbf{B}_2^{-1}\mathbf{P}_0, \text{ а } \mathbf{X}_{j2} = \mathbf{B}_2^{-1}\mathbf{P}_j.$$

Затем подобные же шаги применяются к данным новой таблицы. Заметим, что это преобразование переводит единичную матрицу первоначальной таблицы в матрицу, обратную текущему базису.

Если единичный базис не содержится явно в исходной постановке задачи, то следует ввести в систему множество *искусственных неотрицательных переменных* — по одной переменной на каждое уравнение. В некоторых случаях не требуется полный набор m искусственных переменных, так как часть единичных векторов уже содержится в исходной задаче. Мы предполагаем, что коэффициенты стоимости при искусственных переменных бесконечны, и, следовательно, если существует минимальное допустимое решение исходной задачи, симплекс-метод приведет к значениям искусственных переменных, равным нулю. Если исходная задача не имеет допустимых решений, то симплекс-метод закончится на оптимальном решении, у которого искусственные переменные положительны. Вычислительную таблицу и шаги можно легко модифицировать для случая бесконечных стоимостей.

Чтобы проиллюстрировать все это на конкретном примере, рассмотрим следующую задачу линейного программирования [28]:
максимизировать

$$x_1 + 2x_2$$

при условии, что

$$\begin{aligned}
-x_1 + 3x_2 &\leqslant 10, && \text{(а)}\\
x_1 + x_2 &\leqslant 6, && \text{(б)}\\
x_1 - x_2 &\leqslant 2, && \text{(в)}\\
x_1 + 3x_2 &\geqslant 6, && \text{(г)}\\
2x_1 + x_2 &\geqslant 4, && \text{(д)}\\
x_1 &\geqslant 0,\\
x_2 &\geqslant 0.
\end{aligned}$$

Мы сведем задачу к минимизации, а вместо неравенств получим уравнения, введя свободные переменные x_3, x_4, x_5, x_6, x_7. Поскольку у нас нет полного единичного базиса, мы добавим две искусственные переменные x_8 и x_9. Задача теперь заключается в следующем:

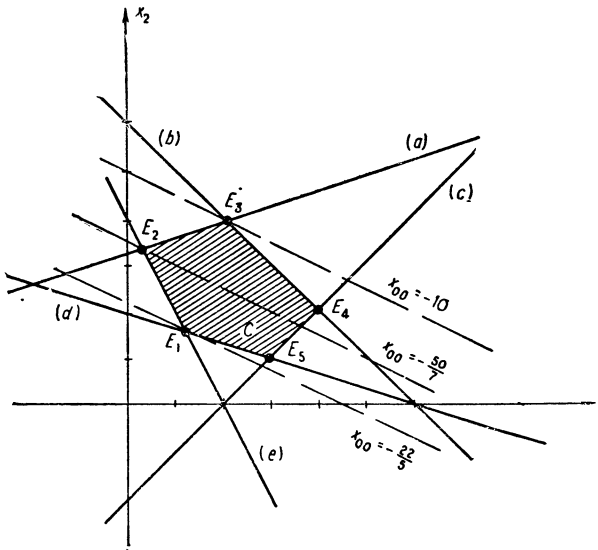
$$
\begin{aligned}
\min = & -x_1 - 2x_2 & & & & & & + wx_8 + wx_9 & \\
& -x_1 + 3x_2 + x_3 & & & & & & = 10, & (\text{а}') \\
& x_1 + x_2 & + x_4 & & & & & = 6, & (\text{б}') \\
& x_1 - x_2 & & + x_5 & & & & = 2, & (\text{в}') \\
& x_1 + 3x_2 & & & - x_6 & & + x_8 & = 6, & (\text{г}') \\
& 2x_1 + x_2 & & & & - x_7 & + x_9 & = 4, & (\text{д}') \\
& & & & & & & x_l \geqslant 0. &
\end{aligned}
$$

w представляет собой бесконечную искусственную стоимость. Первое основное допустимое решение имеет вид $x_3 = 10$, $x_4 = 6$, $x_5 = 2$, $x_8 = 6$, $x_9 = 4$; значение целевой функции равно $10\,w$. Поскольку искусственную часть целевой функции и косвенной стоимости можно отделить от действительной части этих чисел, в таблицу вводится дополнительная строка. Пять таблиц, помещенные на стр. 146, дают полное решение нашей задачи. Обратите внимание, что, пока искусственные переменные присутствуют в решении, в базис следует вводить вектор с максимальной искусственной частью косвенной стоимости. Поскольку искусственные переменные нельзя повторно вводить в базис, мы их удалим из таблиц, как только они исчезнут. Элементы, обведенные кружками,— это опорные элементы. Выполнив последнюю (четвертую) итерацию, мы получили оптимальное решение $x_1 = 2$, $x_2 = 4$, $x_5 = 4$, $x_6 = 8$, $x_7 = 4$, все остальные переменные равны нулю, а максимальное значение целевой функции равно $+10$.

Изображая исходные неравенства в двумерном пространстве, как показано на рисунке, мы получаем геометрическую иллюстрацию симплекс-метода. Заштрихованная область **C** представляет собой выпуклое множество допустимых решений, а точки $\mathbf{E}_i$ суть крайние точки. В данной задаче нам потребовалось два шага симплекс-метода, чтобы определить первое основное

	Базис	P_0	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9
0	P_3	10	−1	3	1						
	P_4	6	1	1		1					
	P_5	2	1	−1			1				
	P_8	6	1	(3)				−1		1	
	P_9	4	2	1					−1		1
		0	1	2							
	w	10	3	4				−1	−1		
1	P_3	4	−2		1			1			
	P_4	4	$\frac{2}{3}$			1		$\frac{1}{3}$			
	P_5	4	$\frac{4}{3}$				1	$-\frac{1}{3}$			
	P_2	2	$\frac{1}{3}$	1				$-\frac{1}{3}$			
	P_9	2	(5/3)					$\frac{1}{3}$	−1		1
		−4	$\frac{1}{3}$					$\frac{2}{3}$			
	w	2	$\frac{5}{3}$					$\frac{1}{3}$	−1		
2	P_3	$\frac{32}{5}$			1			(7/5)	$-\frac{6}{5}$		
	P_4	$\frac{16}{5}$				1		$\frac{1}{5}$	$\frac{2}{5}$		
	P_5	$\frac{12}{5}$					1	$-\frac{3}{5}$	$\frac{4}{5}$		
	P_2	$\frac{8}{5}$		1				$-\frac{2}{5}$	$\frac{1}{5}$		
	P_1	$\frac{6}{5}$	1					$\frac{1}{5}$	$-\frac{3}{5}$		
		$-\frac{22}{5}$						$\frac{3}{5}$	$\frac{1}{5}$		
	w	0									
3	P_6	$\frac{32}{7}$			$\frac{5}{7}$			1	$-\frac{6}{7}$		
	P_4	$\frac{16}{7}$			$-\frac{1}{7}$	1			(4/7)		
	P_5	$\frac{36}{7}$			$\frac{3}{7}$		1.		$\frac{2}{7}$		
	P_2	$\frac{24}{7}$		1	$\frac{2}{7}$				$-\frac{1}{7}$		
	P_1	$\frac{2}{7}$	1		$-\frac{1}{7}$				$-\frac{3}{7}$		
		$-\frac{50}{7}$			$-\frac{3}{7}$				$\frac{5}{7}$		
4	P_6	8			$\frac{1}{2}$	$\frac{3}{2}$		1			
	P_7	4			$-\frac{1}{4}$	$\frac{7}{4}$			1		
	P_5	4			$\frac{1}{2}$	$-\frac{1}{2}$	1				
	P_2	4		1	$\frac{1}{4}$	$\frac{1}{4}$					
	P_1	2	1		$-\frac{1}{4}$	$\frac{3}{4}$					
		−10			$-\frac{1}{4}$	$-\frac{5}{4}$	.				

допустимое решение, соответствующее E_1, затем мы перешли к E_2 и, наконец, к оптимальному решению E_3. На рисунке показаны линии уровня целевой функции, проходящие через каждую из этих крайних точек. В некотором смысле симплекс-метод передвигает линию уровня целевой функции от одной крайней точки к другой, близкой ей крайней точке, пока не будет достигнуто



оптимальное решение. Заметим, что, если бы линии уровня целевой функции были параллельны прямой, соединяющей E_3 и E_4, существовало бы бесконечно много оптимальных решений.

2. Модифицированный симплекс-метод

Нахождение матрицы, обратной текущему базису, представляет собой центральное место вычислительной стороны линейного программирования. Как только найдена эта обратная матрица, можно определить все

величины, участвующие в очередной итерации. Пусть $\mathbf{B}_p$ — допустимый базис p-й итерации. Мы имеем

$$\mathbf{X}_{0p} = \mathbf{B}_p^{-1}\mathbf{P}_0,$$

$$\mathbf{X}_{jp} = \mathbf{B}_p^{-1}\mathbf{P}_j,$$

$$\pi_p = \mathbf{c}_p\mathbf{B}_p^{-1},$$

$$\pi_p\mathbf{P}_j = \mathbf{c}_p\mathbf{B}_p^{-1}\mathbf{P}_j,$$

$$\pi_p\mathbf{P}_0 = \mathbf{c}_p\mathbf{X}_{0p} = \mathbf{c}_p\mathbf{B}_p^{-1}\mathbf{P}_0,$$

где $\mathbf{X}_{0p}$ — вектор, представляющий собой p-е основное допустимое решение; $\mathbf{X}_{jp}$ — векторы, с помощью которых данные векторы $\mathbf{P}_j$ выражаются как линейные комбинации базисных векторов; $\mathbf{c}_p$ — вектор-строка коэффициентов стоимости векторов p-го базиса; элементы вектор-строки π_p называются симплекс-множителями; $\pi_p\mathbf{P}_j$ — косвенная стоимость вектора $\mathbf{P}_j$; $\pi_p\mathbf{P}_0$ — значение целевой функции для p-го базиса. С вычислительной точки зрения использование точного представления обратной матрицы и симплекс-множителей имеет ряд преимуществ. Так, оно позволяет уменьшить количество вычислений и количество информации, которую необходимо записывать при итерациях. В то время как при стандартном симплекс-методе приходится заполнять и преобразовывать всю симплекс-таблицу, при модифицированном варианте требуется записывать только новую обратную матрицу и вектор решения. Отметим, что модифицированный метод использует исходные данные на каждом шаге и что, если, как это часто бывает, они содержат много нулей, экономится время, затрачиваемое на вычисления.

В специальной разновидности модифицированного симплекс-метода используется тот факт, что матрицу, обратную к допустимому базису, совпадающую вначале с единичной матрицей, можно выразить как произведение матриц элементарных преобразований. Каждая такая матрица, которая отличается от единичной матрицы l-м столбцом (l равно номеру вектора, удаляемого из базиса), содержит информацию, необходимую для того,

чтобы определить строку обратной матрицы. Пусть для p-й итерации

$$\mathbf{E}_p^l = \begin{pmatrix} 1 & \dots & y_{1l} & \dots & 0 \\ \cdot & & \cdot & & \cdot \\ \cdot & & \cdot & & \cdot \\ \cdot & & \cdot & & \cdot \\ 0 & \dots & y_{ll} & \dots & 0 \\ \cdot & & \cdot & & \cdot \\ \cdot & & \cdot & & \cdot \\ \cdot & & \cdot & & \cdot \\ 0 & \dots & y_{ml} & \dots & 1 \end{pmatrix},$$

где

$$y_{il} = \frac{x_{ik}}{x_{lk}}, \quad i \neq l;$$

$$y_{ll} = \frac{1}{x_{lk}}.$$

Тогда матрица, обратная p-му базису, равна

$$\mathbf{E}_p^l \mathbf{E}_{p-1}^l \dots \mathbf{E}_2^l \mathbf{E}_1 = \mathbf{B}_p^{-1},$$

где $\mathbf{E}_1 = \mathbf{I}$. При такой сжатой форме записи требуется записывать только ограниченное количество информации. Доказано, что для большинства задач линейного программирования такая разновидность модифицированного симплекс-метода наиболее эффективна. Практика вычислений показала, что число итераций, необходимых для того, чтобы найти оптимальное решение, колеблется между m и $3m$. По-видимому, оно зависит больше от числа уравнений, чем от количества переменных. Оно зависит также от используемого алгоритма, от метода, с помощью которого ищется первое допустимое решение, и от критерия, используемого при выборе вектора, который следует ввести в новый базис.

3. Алгоритм транспортной задачи

Ввиду специальной структуры матрицы коэффициентов, определяющих транспортную задачу, вычисления, которые возникают при использовании в этом случае симплекс-метода, весьма упрощаются. Если m равно числу пунктов отправления, а n — числу пунктов назначения, то каждое допустимое решение транспортной задачи соответствует треугольному базису размерности

$n + m - 1$. Это позволяет сразу же найти решение, а поскольку каждый коэффициент базиса равен 0 или 1, то, если исходные данные выражены в целых числах, и решение выражается в целых числах. Симплекс-алгоритмы для транспортной задачи используют либо решение двойственной задачи, либо специальный метод. Главное различие между ними состоит в способе, с помощью которого вычисляются косвенные стоимости. Двойственный метод используется в большинстве машинных программ. Существуют также разновидности транспортной задачи, в которых используются свои вычислительные методы. Среди них мы отметим задачу, в которой количество грузов, перевозимых между любыми пунктами отправления и назначения, ограничено сверху, а также обобщенную транспортную задачу, или задачу о загрузке машин, в которой коэффициенты уравнений могут быть отличными от 0 или 1.

4. Дополнительные вычислительные приемы

а. Двойственный симплекс-алгоритм

Иногда бывает легче найти базис, удовлетворяющий критерию оптимальности (все $x_{0j} \leqslant 0$), но не удовлетворяющий критерию допустимости (не все $x_{i0} \geqslant 0$). В этом случае можно использовать двойственный симплекс-алгоритм. Единственное отличие его состоит в критерии, используемом при выборе того вектора (переменной), который следует ввести в базис, и того, который следует удалить из базиса. В данном случае сперва определяется вектор, который следует удалить, соответствующий

$$x_{l0} = \min x_{i0} < 0.$$

Новый вектор соответствует индексу k, для которого

$$\frac{x_{0k} - c_k}{x_{lk}} = \min_{x_{lj} < 0} \frac{x_{0j} - c_j}{x_{lj}} > 0.$$

Опорным элементом является x_{lk}, а преобразование, с помощью которого удаляется вектор, то же, что и в стандартном симплекс-методе. При этом хотя бы одна из отрицательных переменных становится положительной, а преобразованные x_{0j} по-прежнему удовлетворяют условию $x_{0j} \leqslant 0$, то есть данное преобразование улуч-

шает «допустимость» решения, не нарушая в то же время критерий оптимальности. Первоначальный симплекс-метод (исходный метод) и двойственный метод комбинируют иногда в *исходно-двойственных*, или *симплекс-составных*, машинных программах. При использовании этих приемов не накладывается ограничений на знаки c_j или b_j, и с их помощью можно получать исходное основное решение, удовлетворяющее либо условиям допустимости, либо условиям оптимальности.

б. Программирование в целых числах

Во многих задачах линейного программирования требуется, чтобы решение выражалось в целых числах, например, если нам нужно найти целое число единиц данной продукции. В отличие от транспортной задачи симплекс-алгоритм не гарантирует, что решение задачи линейного программирования общего вида будет выражаться в целых числах. Однако существуют разновидности симплекс-метода, гарантирующие, что оптимальное решение, если оно существует, будет выражаться в целых числах. Этого можно достигнуть, добавляя систематическим образом новые ограничения, или секущие плоскости, к данному множеству ограничений. Новые ограничения изменяют выпуклое множество решений так, что оптимальной крайней точкой этого множества становится точка с целыми координатами. Существуют и соответствующие машинные программы, однако ввиду того, что число итераций, необходимых для решения некоторой задачи, существенно зависит от структуры и исходных данных этой задачи, вопрос об использовании таких программ в операционных исследованиях все еще остается открытым.

в. Условия ограниченности сверху

Во многих случаях переменные, встречающиеся в задачах линейного программирования, ограничены сверху, то есть $x_j \leqslant u_j$. И снова небольшое изменение основного симплекс-метода позволяет решать задачи с такими ограничениями, не вводя эти ограничения в симплекс-таблицу. Данный метод можно использовать, когда ограничены все или только некоторые переменные. Со случаем, когда x_j ограничены снизу, то есть $d_j \leqslant x_j$,

можно легко справиться, применив прямую подстановку. Мы полагаем $x_j = d_j + x'_j$ и подставляем $d_j + x'_j$ вместо соответствующих x_j.

г. Анализ устойчивости

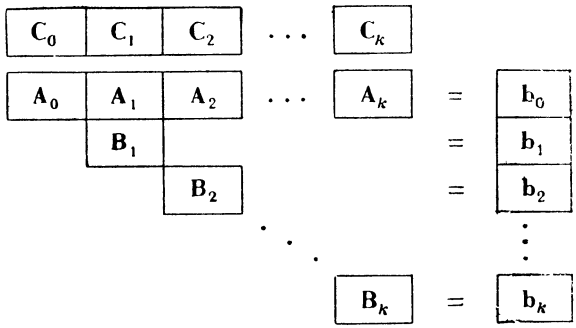
Анализ устойчивости в линейном программировании занимается вопросом о том, как влияют изменения исходных данных на оптимальное решение. Например, нас интересует, в каких пределах можно изменять коэффициенты стоимости, прежде чем оптимальное решение перестанет быть таковым; или на сколько можно изменить коэффициенты b_i, прежде чем решение перестанет быть допустимым; или, наконец, как влияет на оптимальное решение изменение a_{ij}. Есть приемы, которые позволяют ответить на все эти вопросы и которые включаются обычно в соответствующие машинные программы. Существуют также методы, предназначенные для случаев параметрического программирования, когда коэффициенты стоимости и (или) свободные члены зависят линейным образом от одного и того же параметра, например $c_j = d_j + \lambda d'_j$. С помощью этих методов находятся решения, соответствующие различным значениям параметра.

д. Алгоритм разбиения

Хотя теоретически возможно найти решение для любой модели линейного программирования, аналитик быстро осознает, что существуют некоторые ограничения, стоящие на пути его усилий. Главным среди этих ограничений является проблема размерности. Почти все трудности, возникающие при решении задач линейного программирования, так или иначе с ней связаны. Это, безусловно, верно для вопросов, касающихся стоимости сбора данных, составления матрицы, вычисления стоимостей, разумности линейной модели и т. д.

Во многих задачах ограничения состоят из довольно больших подмножеств уравнений, которые имеют между собой то общее, что относятся к одному периоду времени или к одним и тем же производственным возможностям. Эти подмножества связаны между собой небольшим числом уравнений. Такие «уравнения связи» могут,

например, выражать общую потребность в каком-либо продукте. В задачах подобного рода мы в некотором смысле имеем целый ряд отдельных задач линейного программирования, объединенное решение которых должно удовлетворять некоторым дополнительным ограничениям. Если мы разобьем множество ограничений и целевую функцию на блоки, то получим



Здесь мы свели исходную задачу (найти минимум $\mathbf{cX}$ при условии $\mathbf{AX} = \mathbf{b}$, $\mathbf{X} \geqslant \mathbf{0}$) к совокупности более мелких задач: найти векторы $\mathbf{X}_p \geqslant \mathbf{0}$ ($p = 0, 1, \ldots, k$), такие, чтобы

$$\sum_p \mathbf{A}_p\mathbf{X}_p = \mathbf{b}_0, \tag{8}$$

$$\mathbf{B}_p\mathbf{X}_p = \mathbf{b}_p, \tag{9}$$

а

$$\sum_p \mathbf{C}_p\mathbf{X}_p \text{ достигала минимума;} \tag{10}$$

здесь $\mathbf{A}_p$ — матрица $(m_0 \times n_p)$; $\mathbf{B}_p$ — матрица $(m_0 \times n_p)$; $\mathbf{C}_p$ — n_p-мерный вектор; $\mathbf{b}_0$ — m_0-мерный вектор; $\mathbf{b}_p$ — m_p-мерный вектор; $\mathbf{X}_p$ — n_p-мерный вектор, составленный из неизвестных. Как мы видим, в этой задаче $\sum_p m_p$ ограничений и $\sum_p n_p$ переменных. Мы сейчас предполагаем, что в нашей задаче существует конечный оптимум.

Предположим также, что для каждого p существует соответствующее выпуклое множество $\mathbf{S}_p$ решений каждой из меньших задач $\mathbf{B}_p\mathbf{X}_p = \mathbf{b}_p$, $\mathbf{X}_p \geqslant \mathbf{0}$. Тогда решение исходной задачи можно представлять себе как некоторую выпуклую комбинацию решений из $\mathbf{S}_p$,

удовлетворяющую ограничениям связи $\sum_p \mathbf{A}_p\mathbf{X}_p=\mathbf{b}_0$ и доставляющую минимум выражению $\sum_p \mathbf{C}_p\mathbf{X}_p$. Хотя эта голая идея разбиения кажется чреватой своими собственными трудностями, она дает нам некую экономию, поскольку мы можем рассматривать задачу об оптимизации с $m_0 + k$ ограничениями при условии, что найдены решения $k(m_p \times n_p)$ меньших задач, вместо того чтобы решать большую задачу с $\sum_p m_p$ ограничениями.

Рассмотрим новую задачу, называемую экстремальной программой или главной задачей, которая возникает следующим образом. Возьмем какую-нибудь крайнюю точку $\mathbf{X}_{pj}$ выпуклого множества решений $\mathbf{S}_p$ задачи $\mathbf{X}_p \geqslant \mathbf{0}$, $\mathbf{B}_p\mathbf{X}_p = \mathbf{b}_p$ при некотором p. Определим для каждой такой крайней точки j

$$\mathbf{P}_{pj} = \mathbf{A}_p\mathbf{X}_{pj}, \quad f_{pj} = \mathbf{C}_p\mathbf{X}_{pj}.$$

Экстремальная программа состоит в том, чтобы найти числа $\lambda_{pj} \geqslant 0$, удовлетворяющие для всех (p, j)

$$\mathbf{A}_0\mathbf{X}_0 + \sum_p \sum_j \mathbf{P}_{pj}\lambda_{pj} = \mathbf{b}_0, \tag{11}$$

$$\sum_j \lambda_{pj} = 1 \quad (p = 1, 2, \ldots, k), \tag{12}$$

и такие, что

$$\mathbf{C}_0\mathbf{X}_0 + \sum_p \sum_j f_{pj}\lambda_{pj} \text{ минимально.} \tag{13}$$

Как отмечают в одной из своих работ Данциг и Вольф, «связь между экстремальной и исходной задачами выражается в том факте, что любая точка $\mathbf{S}_p$, поскольку оно представляет собой ограниченный (по предположению) и выпуклый многогранник, может быть выражена как линейная комбинация его крайних точек, то есть в виде $\sum_j \mathbf{X}_{pj}\lambda_{pj}$, где λ_{pj} удовлетворяют (12), а выражения (11) и (13) совпадают в точности с выражениями (8) и (10), переписанными в терминах λ_{pj}».

Как указывалось выше, это преобразование приводит к задаче только с $m_0 + k$ ограничениями [m_0 ограничений связи (11) и k ограничений (12)]. Однако число переменных возрастает и становится равным общему количеству крайних точек выпуклого множества $\mathbf{S}_p$, очень

большой величине. Экономия, которой мы достигаем, применяя метод разбиения, заключается в том, что нам приходится рассматривать только небольшую часть этого общего количества и что нам нужно точное представление только для тех точек, которые мы рассматриваем, и только тогда, когда требуется.

IV. ПЕРЕЧЕНЬ ПРИЛОЖЕНИЙ ЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ

В этом разделе дается обзор материала I—IV глав, более формально излагаются некоторые важные приложения линейного программирования, а также, где это возможно, строятся типичные математические модели.

Надо подчеркнуть, что такие модели, важные сами по себе, следует рассматривать только как исходные позиции или основные модели при подходе к формулировке подобных задач. Может случиться, что задача, с которой столкнется читатель, окажется похожей на одну из наших моделей, и тогда при надлежащей модификации основной модели возможно удастся получить хорошее первое приближение к реальной ситуации. Прекрасным примером в этом отношении служит транспортная задача, которую следует скомбинировать с моделью производства и хранения для того, чтобы отразить истинное положение дел. Нужно остерегаться искусственной подгонки задачи под некоторую модель и в то же время помнить, что уже сам выбор основной модели может привести к эволюции наших представлений о том, что происходит в действительности.

1. Транспортные задачи

а. Основная транспортная задача

Требуется доставить однородный продукт из m пунктов отправления в количествах $a_1, a_2, \ldots, a_m$ соответственно в n пунктов назначения в количествах $b_1, b_2, \ldots, b_n$ соответственно. Стоимость c_{ij} доставки одной единицы продукта из пункта i в пункт j известна для всех комбинаций (i, j). Задача состоит в том, чтобы определить количество продукта x_{ij}, которое нужно отправить по каждому пути (i, j), для того чтобы минимизировать общую стоимость доставки. a_i представляют собой наличные, а b_j — требуемые количества продукта.

Чтобы сделать математическую модель более содержательной, мы потребуем выполнения равенства $\sum_i a_i = \sum_j b_j$, то есть общее наличное количество продукта равно общему требуемому количеству. Это ограничение не обязано присутствовать в каждом конкретном приложении, поскольку, если $\sum_i a_i > \sum_j b_j$, к задаче добавляется фиктивный пункт назначения (например, излишки скапливаются на складах) с требуемым количеством $\sum_i a_i - \sum_j b_j$. Если $\sum_i a_i < \sum_j b_j$, то к задаче добавляется фиктивный пункт отправления (например, закупки у конкурента) с наличным количеством $\sum_j b_j - \sum_i a_i$.

Пусть $x_{ij} \geqslant 0$ — неизвестное количество, которое следует отправить из пункта i в пункт j (то есть x_{ij} — множество переменных). Тогда модель линейного программирования для случая $m = 2$, $n = 3$ следующая:
минимизировать

$$c_{11}x_{11} + c_{12}x_{12} + c_{13}x_{13} + c_{21}x_{21} + c_{22}x_{22} + c_{23}x_{23}$$

при условии, что

$$\begin{aligned}
x_{11} + x_{12} + x_{13} &= a_1,\\
x_{21} + x_{22} + x_{23} &= a_2,\\
x_{11} + x_{21} &= b_1,\\
x_{12} + x_{22} &= b_2,\\
x_{13} + x_{23} &= b_3,\\
x_{ij} &\geqslant 0.
\end{aligned}$$

Поскольку любое из написанных выше уравнений можно вывести из $(m + n - 1)$ остальных, ранг системы равен $(m + n - 1)$. Каждый допустимый базис имеет порядок $(m + n - 1)$, а соответствующая матрица будет треугольной. У данной модели есть широкий круг приложений, в которых пункты отправления и назначения получают весьма разнообразную интерпретацию (например, склады, магазины, порты), а коэффициенты стоимости представляют собой расстояния, время, доллары и т. д. Эту модель можно модифицировать, например, чтобы получить информацию, позволяющую выбрать из многих новых возможных пунктов отправления один, с

точки зрения транспортных цен самый лучший. Здесь приходится решать целое множество задач, по одной на каждую возможность, с данными, отражающими ожидаемые стоимости перевозок между новым пунктом отправления и старыми пунктами назначения, а также ожидаемое наличие продукта в новом пункте и требуемые его количества в старых пунктах. Аналогичный анализ мог бы помочь при выборе нового пункта назначения, а также при выборе того пункта отправления или назначения, который следует закрыть.

б. Транзитная задача

Это транспортная задача, в ней пункты отправления и назначения могут выступать в качестве промежуточного пункта, через который переправляются товары в конечный пункт назначения. Данную задачу можно свести к основной транспортной задаче и решать методами, применимыми к последней.

в. Транспортная задача с ограниченными возможностями

Это основная транспортная задача, в которой возможные перевозки ограничены сверху, то есть $x_{ij} \leqslant u_{ij}$. Такую задачу можно решить с помощью небольшой модификации того приема, который применяется при решении основной задачи. Существует также специальный исходно-двойственный алгоритм.

г. Обобщенная транспортная задача и задача о распределении станков

В данном случае постановка задачи такова: минимизировать

$$\sum_i \sum_j c_{ij} x_{ij}$$

при условии, что

$$\sum_{j=1}^{n} a_{ij} x_{ij} = a_i \quad (i = 1, \ldots, m), \tag{14}$$

$$\sum_{i=1}^{m} b_{ij} x_{ij} = b_j \quad (j = 1, \ldots, n). \tag{15}$$

$$x_{ij} \geqslant 0$$

Задачи, для которых подходит данная модель, можно найти в области транспорта и среди задач о распределении станков. В последнем случае $b_{ij} = 1$, уравнения (14) превращаются в неравенства ($\leqslant$); a_{ij} представляют собой время, затрачиваемое на производство одной единицы продукта j на станке i; x_{ij} — число единиц продукта j, изготовляемое на станке i; b_j — число единиц j, которое необходимо получить; c_{ij} — стоимость производства одной единицы продукта j на станке i.

д. Многомерная транспортная задача

Модель этой задачи бывает двух видов:

1) Минимизировать

$$\sum_i \sum_j \sum_k c_{ijk} x_{ijk}$$

при условии, что

$$\sum_i x_{ijk} = a_{jk},$$

$$\sum_j x_{ijk} = b_{ik},$$

$$\sum_k x_{ijk} = c_{ij},$$

$$x_{ijk} \geqslant 0.$$

2) Минимизировать

$$\sum_i \sum_j \sum_k c_{ijk} x_{ijk}$$

при условии, что

$$\sum_j \sum_k x_{ijk} = a_i,$$

$$\sum_i \sum_k x_{ijk} = b_j,$$

$$\sum_i \sum_j x_{ijk} = d_k,$$

$$x_{ijk} \geqslant 0.$$

2. Задачи о распределении или назначении

В задачах этого типа нам дано определенное количество индивидуумов, машин и т. д., которое требуется распределить для выполнения некоего множества работ.

Для каждого индивидуума i задана оценка c_{ij}, измеряющая его эффективность при выполнении работы j. Индивидуум может быть назначен только на одну работу. Если x_{ij} выражает собой назначение i-го индивидуума на j-ю работу, то формулировка линейного программирования такова:
максимизировать

$$\sum_i \sum_j c_{ij} x_{ij}$$

при условии, что

$$\sum_j x_{ij} = a_i \quad (i = 1, \ldots, m),$$

$$\sum_i x_{ij} = b_j \quad (j = 1, \ldots, n),$$

$$x_{ij} \geqslant 0,$$

где a_i — наличное количество индивидуумов типа i, а b_j — наличное количество работ типа j. Мы полагаем, что $\sum a_i = \sum b_j$. Во многих случаях все a_i и b_j равны 1 и $m = n$. С математической точки зрения задача о назначении представляет собой транспортную задачу.

3. Задача о танкерных перевозках

Дано некоторое число портов $i = 1, 2, \ldots, m$, в которых производится загрузка танкеров, отправляющихся затем в порты назначения $j = 1, 2, \ldots, n$. Мы знаем время, необходимое для загрузки в порту i танкера, отправляющегося в порт j, а также время, затрачиваемое на путь из одного порта в другой. (Мы предполагаем, что все танкеры одинаковы, а перевозки выражаются в единицах вместимости танкера.) Доставив груз, танкер может идти в любой погрузочный порт. Задача состоит в том, чтобы определить такой маршрут для каждого танкера, при котором общее количество танкеров будет минимальным.

4. Сетевые задачи о потоке

Задана некоторая сеть (шоссе, железные дороги, трубопроводы и т. д.), содержащая единственный источник (пункт отправления), единственный сток (пункт назначения) и промежуточные узлы (передаточные пунк-

ты). Пусть x_{ij} — поток из пункта i в пункт j нашей сети. Пункт может быть либо источником, либо стоком, либо промежуточным узлом. Мы предполагаем, что сеть ориентирована и что для каждого пути (i, j) заданы ограничения на его пропускную способность $f_{ij} \geqslant 0$. Мы хотим так распределить поток вдоль сети, чтобы общий поток f из источника в сток был максимальным.

Для сети общего вида с источником $i = 0$ и стоком $i = m$ мы получаем следующую формулировку:
максимизировать

$$f$$

при условии, что

$$\sum_i (x_{0i} - x_{i0}) = f, \tag{16}$$

$$\sum_j (x_{ij} - x_{ji}) = 0, \quad i \neq 0, m, \tag{17}$$

$$\sum_i (x_{mi} - x_{im}) = -f, \tag{18}$$

$$0 \leqslant x_{ij} \leqslant f_{ij}. \tag{19}$$

Равенство (17) выражает собой условие сохранения потока в промежуточных узлах, то есть то, что втекает в данный узел, должно и вытекать из него, а суммирование ведется по тем j, которые непосредственно связаны путями с данным i. Целевая функция максимизирует поток из источника или поток в сток — (16) и (18).

5. Заключение контрактов

Когда какому-либо правительственному учреждению требуется закупить некоторые товары, оно должно пригласить фирмы, производящие их, и принять участие в заключении контрактов. Каждая фирма представляет свои предложения, отражающие ее желание получить прибыль, ее догадки относительно предложений конкурентов и ее собственные ограничения. Покупатель должен заключить контракты таким образом, чтобы общая стоимость затрат была минимальна. Задачи такого типа можно свести к последовательности транспортных задач и решить с помощью этого алгоритма.

6. Задачи об анализе деятельности

Некий производитель имеет в своем распоряжении определенное количество различных ресурсов. Эти ресурсы, такие, как сырой материал, труд и оборудование, комбинируются для производства некоторых товаров или комбинаций товаров. Предпринимателю известно, сколько требуется ресурса i для того, чтобы получить единицу товара j. Ему также известно, какой доход он получит от каждой единицы изготовленного товара j. Предприниматель хочет изготовить такую комбинацию товаров, которая максимизировала бы его общий доход. Введем следующие обозначения:

m — число ресурсов;

n — число товаров;

a_{ij} — число единиц ресурса i, требуемое для производства одной единицы товара j;

b_i — максимальное наличное количество единиц ресурса i;

c_j — доход на единицу изготовленного товара j;

x_j — уровень производства (произведенное количество) j-го товара.

Коэффициенты a_{ij} называют иногда коэффициентами затраты — выпуска.

Общее количество затраченного i-го ресурса дается линейным выражением

$$a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{in}x_n.$$

Так как это общее количество не должно превышать максимальное число наличных единиц i-го ресурса, мы получаем для каждого i линейное неравенство вида

$$a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{in}x_n \leqslant b_i.$$

Поскольку для отрицательных x_j нет подходящей интерпретации, мы требуем, чтобы все $x_j \geqslant 0$. Доход, который приносят x_j единиц j-го товара, равен c_jx_j. Задача формулируется следующим образом:
максимизировать функцию доходов

$$c_1x_1 + c_2x_2 + \ldots + c_nx_n$$

при условии, что

$$\begin{aligned}
a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n &\leqslant b_1\\
a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n &\leqslant b_2\\
\cdot \quad\quad \cdot & \quad\quad \cdot\\
\cdot \quad\quad \cdot & \quad\quad \cdot\\
\cdot \quad\quad \cdot & \quad\quad \cdot\\
a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n &\leqslant b_m
\end{aligned}$$

и

$$\begin{aligned}
x_1 &\geqslant 0\\
x_2 &\geqslant 0\\
\cdot &\quad \cdot\\
\cdot &\quad \cdot\\
\cdot &\quad \cdot\\
x_n &\geqslant 0
\end{aligned}$$

Эти задачи возникают в области экономики при обсуждении теории фирм и межотраслевого анализа (затраты — выпуск).

7. Задача о диете

Здесь нам задан состав некоторых продуктов питания. Например, нам может быть известно, сколько миллиграммов фосфора или железа содержится в одной унции каждого из рассматриваемых продуктов. Мы знаем также минимальную дневную потребность в каждом из необходимых для поддержания жизнедеятельности веществ. Так как цена одной унции каждого продукта питания известна, задача заключается в том, чтобы определить диету, удовлетворяющую дневным минимальным потребностям в белках, жирах, витаминах и т. д. и обладающую минимальной стоимостью. Пусть

m — число необходимых веществ;

n — число продуктов питания;

a_{ij} — число миллиграммов i-го вещества в одной унции j-го продукта питания;

b_i — минимальное количество i-го вещества, необходимое для нормальной деятельности организма;

c_j — стоимость одной унции j-го продукта питания;

x_j — число унций j-го продукта, которое следует закупить ($x_j \geqslant 0$).

Общее количество i-го вещества во всей покупке задается выражением

$$a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{in}x_n.$$

Поскольку это количество должно быть больше или равно минимальной дневной потребности в i-ом веществе, данная задача заключается в том, чтобы минимизировать функцию стоимости

$$c_1x_1 + c_2x_2 + \ldots + c_nx_n$$

при условии, что

$$\begin{aligned}
a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n &\geqslant b_1,\\
a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n &\geqslant b_2,\\
\cdot \quad\quad \cdot \quad\quad\quad\quad \cdot \quad\quad \cdot &\\
\cdot \quad\quad \cdot \quad\quad\quad\quad \cdot \quad\quad \cdot &\\
\cdot \quad\quad \cdot \quad\quad\quad\quad \cdot \quad\quad \cdot &\\
a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n &\geqslant b_m
\end{aligned}$$

и

$$\begin{aligned}
x_1 \quad\quad &\geqslant 0\\
x_2 \quad\quad &\geqslant 0\\
\cdot \quad\quad &\cdot\\
\cdot \quad\quad &\cdot\\
\cdot \quad\quad &\cdot\\
x_n &\geqslant 0
\end{aligned}$$

Хотя эта формулировка и спорна в применении к диете, рассчитанной на людей, она оказалась весьма удовлетворительной при составлении диет для крупного рогатого скота и цыплят.

8. Задача о составлении смесей

Подобные задачи возникают в ситуациях, когда один или более продуктов получаются путем смешивания некоторого числа компонентов. Существуют ограничения на количество имеющихся в наличии видов сырья, ограничения на количество и качество производимых продуктов. Обычно можно бесконечным числом способов смешать различные виды сырья, чтобы получить конечные продукты, удовлетворяющие различным ограничениям. Мы хотим так составить смеси, чтобы оптимизировать заданную целевую функцию. Задачи о смесях встречаются в нефтехимической промышленности, при произ-

водстве красок, стали и т. д. Для того чтобы описать соответствующую математическую модель, нам пришлось бы вникать в детали конкретных процессов производства, что выходит за рамки настоящей книги. Интересующегося этим читателя мы отсылаем к библиографии.

9. Планирование производства

Предприниматель знает, что ему надо изготовить r_i ($i = 1, 2, \ldots, n$) единиц некоторого товара в течение ближайших n месяцев. Это количество можно производить или регулярным образом, не более a_i единиц в месяц, или форсированным образом, не более b_i единиц в месяц. Стоимость производства одной единицы товара в i-м месяце равна c_i при регулярном и d_i при форсированном способе производства. Ввиду колебаний стоимости с течением времени, а также ввиду ограниченных возможностей может оказаться, что экономичнее производить товар впрок, до того как он потребуется в действительности. Предположим, что стоимость хранения единицы товара в течение месяца равна s_i. Мы хотим так спланировать производство, чтобы минимизировать общую сумму затрат на производство и хранение. Хотя эту задачу можно сформулировать как стандартную задачу линейного программирования, ее также можно сформулировать в терминах транспортной задачи и решить с помощью этого алгоритма.

10. Сглаживающие схемы производства

Производитель некоторого продукта должен спланировать свое производство на ближайшие n месяцев. Потребность в его продукте меняется, однако он должен постоянно удовлетворять месячные заказы как сделанные заранее, так и ожидаемые. Он может удовлетворить индивидуальные заказы, производя либо требуемое количество продукта в течение данного месяца, либо только часть его, а недостачу восполняя за счет излишков продукции, произведенной в предыдущие месяцы.

Вообще говоря, в любой задаче планирования существует много разных планов, удовлетворяющих заданным требованиям. Например, предприниматель может производить каждый месяц точно то количество, которое, как ожидается, будет заказано. Однако такой тип плана не эффективен, поскольку придерживаться его

слишком накладно. С другой стороны, наш предприниматель мог бы производить излишки продукции, когда на нее небольшой спрос, и воспользоваться ими в период высокого спроса. Схему производства, следовательно, можно сделать весьма стабильной. Но, ввиду того что на хранение готовой продукции тратятся средства, такое решение может оказаться нежелательным, если оно приводит к сравнительно большим дополнительным месячным издержкам. В этой задаче мы хотим так спланировать производство, чтобы минимизировать сумму издержек, вызванных флуктуациями выпуска и хранением.

Пусть

x_t — продукция в месяце t;

r_t — потребности в месяце t;

s_t — количество продукта на складе в месяце t.

Формулировка нашей задачи принимает следующий вид: минимизировать

$$a \sum_t y_t + b \sum_t s_t$$

при условии, что

$$x_t + s_{t-1} - s_t = r_t, \tag{20}$$

$$x_t - x_{t-1} - y_t + z_t = 0, \tag{21}$$

где $(x_t, s_t, y_t, z_t) \geqslant 0$; (20) означает, что количество продукта, изготовленное в месяце t, плюс количество, хранившееся на складе в предыдущем месяце, равно месячным потребностям и количеству, хранившемуся на складе в месяце t; (21) означает, что разность между продукцией в месяце t и продукцией в месяце $(t-1)$ можно представить как разность двух неотрицательных чисел, где y_t — увеличение продукции, а z_t — ее уменьшение; a — стоимость увеличения продукции на единицу, а b — стоимость хранения единицы продукции на складе в течение месяца.

11. Задача о потерях

Бумажные фабрики производят рулоны заданной стандартной ширины. Заказчикам требуются рулоны различной ширины, и, следовательно, стандартные рулоны нужно разрезать. Вообще говоря, в процессе разрезания образуются некоторые потери. Предприниматель хочет нарезать рулоны таким образом, чтобы, с одной стороны, удовлетворить требования заказчиков, а с

другой — минимизировать общие потери. Эту же схему можно применить к аналогичным производственным ситуациям, в которых требуется разрезать стандартные рулоны, листы и т. п., причем образуются потери.

12. Задача поставщика

Поставщик кулинарных изделий знает, что ему потребуются в течение ближайших n дней свежие салфетки по r_j штук в день, $j = 1, 2, \ldots, n$. Стирка обычно занимает p дней, то есть если грязная салфетка посылается в прачечную немедленно после того, как была использована в j-й день, то она возвращается назад и вновь может быть использована в $(j + p)$-й день. Кроме того, в прачечной есть срочное обслуживание, при котором салфетки возвращаются через $q < p$ дней (p и q — целые числа). Не имея под рукой или в прачечной нужного количества салфеток, поставщик удовлетворяет первоначально свои потребности, покупая салфетки по a центов за штуку. Стирка одной салфетки стоит при обычном обслуживании b, а при срочном c центов. Как следует поступать поставщику, чтобы удовлетворить свои нужды и минимизировать расходы в данные n дней?

13. Задача коммивояжера

Задача состоит в том, чтобы найти кратчайший путь для коммивояжера, который отправляется из данного города, посещает указанную группу городов и возвращается назад, в исходный пункт. При формулировке данной задачи линейного программирования требуется, чтобы переменные выражались в целых числах.

14. Складская задача

Пусть задан склад фиксированной вместимости и начальный запас некоторого продукта, подверженный сезонному колебанию цены и стоимости. Пусть, кроме того, задано время задержки между покупкой и получением данного продукта. Какой схемы покупок, хранения и продажи следует придерживаться, чтобы максимизировать доход в данный период времени?

15. Сетевое планирование

Характерной чертой многих проектов является то, что вся работа должна выполняться в некотором хорошо определенном порядке; например, при строительных работах следует вначале сделать опалубку, а уж потом заливать бетон. Целью данной формулировки является планирование работ, составляющих проект. При таком анализе требуется графическое представление проекта, при котором известны стоимость и время начала и окончания каждой работы. Формулировка линейного программирования дает возможность выбрать схему с наименьшей стоимостью для желаемого и допустимого времени выполнения проекта.

16. Расчеты и конструирование

Класс задач, посвященный расчетам и конструированию, например расчетам строительных конструкций, допускает трактовку с точки зрения линейного программирования. Но, поскольку данные вопросы носят слишком специальный характер, мы отсылаем интересующегося читателя к библиографии.

17. Игры двух партнеров с нулевой суммой

Следует отметить, что любую игру двух партнеров с нулевой суммой можно представить как задачу линейного программирования, а задачу линейного программирования тоже можно преобразовать в игру двух партнеров с нулевой суммой. В случае игры формулировка, двойственная формулировке линейного программирования для одного из игроков, представляет собой формулировку линейного программирования для другого игрока. Вычислительные методы одной из этих областей можно применять при решении задачи, относящейся к другой области

ЛИТЕРАТУРА

Линейное программирование

1. *Ackoff R. L.*, The Development of Operations Research as a Science, J. Oper. Res. Soc. Am., **4**, 3 (June 1956).
2. *Allen R. G. D.*, Mathematical Economics, N. Y., Macmillan, 1956.
3. *Bellmore M., Nemhauser G. L.*, The Travelling Salesman Problem: A Survey, Oper. Res., **16**, 3 (June 1968).
4. *Charnes A., Cooper W. W.*, Management Models and Industrial Application of Linear Programming, N. Y., Wiley, 1960.
5. *Charnes A., Cooper W. W.*, The Stepping Stone Metod of Explaining Linear Calculations in Transportation Problems, Manag. Sci., **1** (1954—1955).
6. *Charnes A., Cooper W. W., Henderson A.*, Introduction to Linear Programming, N. Y., Wiley, 1953.
7. *Charnes A., Cooper W. W., Mellon B.*, Blending Aviation Gasolines — A Study in Programming Interdependent Activities in an Integrated Oil Company, Econometrica, **20** (1952).
8. *Churchman C. W., Ackooff R. L., Arnoff E. L.*, Introduction to Operations Research, N. Y., Wiley, 1957.
9. *Cooper W. W., Charnes A.*, Linear Programming, Sci. Am. (August 1954).
10. *Dantzig G. B.*, Maximisation of a Linear Function of Variables Subject to Linear Inequalities, см. в книге [27].
11. *Dantzig G. B.*, Application of the Simplex Method to a Transportation Problem, см. в книге [27].
12. *Dantzig G. B.*, Linear Programming, Princeton, Princeton Univ. Press, 1963.
13. *Dennis J. B.*, A High-speed Computer Technique for the Transportation Problem, J. Ass. Comput. Mach., **5**, 2 (1958).
14. *Dicson J. C., Frederic F. P.*, A Decision Rule for Improved Efficiency in Solving Linear Problems with the Simplex Algorithm, Communicat. ACM, **3**, 9 (1960).
15. *Dorfman R.*, Application of Linear Programming to the Theory of the Firm, Berkley, Univ. Calif. Press, 1951.
16. *Dorfman R.*, Mathematical, or «Linear», Programming, Am. Econom. Rev., **43** (December 1953).
17. *Dorfman R., Samuelson P. A., Solow R.*, Linear Programming and Economic Analysis, N. Y., McGraw-Hill, 1958.

18. *Eisemann K., Lourie J. R.*, The Machine Loading Problem, IBM Applications Library, N. Y., 1959.
19. *Gale D.*, The Theory of Linear Economic Models, N. Y., McGraw-Hill, 1960.
20. *Garvin W. W.*, Introduction to Linear Programming, N. Y., McGraw-Hill, 1960.
21. *Gass S. I.*, Recent Developments in Linear Programming, Advances in Computers, N. Y., Acad. Press, **2**, 1961.
22. *Good H. W.*, The Application of a Highspeed Computer to the Definition and Solution of the Vehicular Traffic problem, J. Oper. Res. Soc., **5**, 6 (December 1957).
23. *Hadley G.*, Linear Programming, Reading, Addison-Wesley, 1962.
24. *Hayman J., Prager W.*, Automatic Minimum Weight Design of Steel Frames, J. Franklin Inst., **266**, 5 (1958).
25. *Kelley J. E.*, Jr., The Cutting-plane Method for Solving Convex Programs, Ambler, Mauchly Ass., 1959.
26. *Kelley J. E., Jr.*, Parametric Programming and Primal-Dual Algorithm, Oper. Res., **7**, 3 (1959).
27. *Koopmans T. C.* (ed.), Activity Analysis of Production and Allocation, Cowles Commission Monograph 13, N. Y., Wiley, 1951.
28. *Lemke C. E.*, The Dual Method of Solving the Linear Programming Problem, Nav. Res. Logist Quart., **1**, 1 (1954).
29. *Lemke C. E., Charnes A., Sienkiewicz O. C.*, Plastic Limit Analysis and Integral Linear Programs, Mathematical Report 21, Troy, Rensselaer Polytechn. Inst., 1959.
30. *Luce R. D., Raiffa H.*, Games and Decisions, N. Y., Wiley, 1957.
31. *Мак-Кинси Дж.*, Введение в теорию игр, М., Физматгиз, 1960.
32. *Riley V., Gass S. I.*, Bibliography on Linear Programming and Related Techniques, Baltimore, Johns Hopkins Press, 1958.
33. *Saaty T. L.*, Mathematics Methods of Operations Research, N. Y., McGraw-Hill., 1959.
34. *Tucker A. W.*, An Integer Program for a Multiple-trip Variant of the Travelling Salesman Problem, Princeton, Princeton Univ., 1960.
35. *Vajda S.*, Mathematical Programming, Reading, Addison-Wesley, 1958.
36. *Vajda S.*, Readings in Mathematical Programming, N. Y., Wiley, 1962.
37. *Wagner H. M.*, A Comparison of the Original and Revised Simplex Methods, Oper. Res., **5**, 3 (1957).
38. *Wagner H. M.*, A Practical Gide to the Dual Theorem, Oper. Res., **6**, 3 (1958).
39. *Wagner H. M.*, The Simplex Method for Beginners, Oper. Res., **6**, 2 (1958).
40. *Wagner H. M.*, The Dual Simplex Algorithm for Bounded Variables, Nav. Res. Logist. Quart., **5**, 3 (1958).
41. *Wolf P.*, The Composite Simplex Algorithm, SIAM Rev., **7**, 1 (1965).
42. *Гасс С. И.*, Линейное программирование (методы и приложения), М., Физматгиз, 1961.
43. *Грешер М.*, Стратегические игры. Теория и приложения, М., изд-во «Советское радио», 1964.
44. Линейные неравенства, Сб. переводов, М., ИЛ, 1959.
45. *Юдин Д. Б., Гольштейн Е. Г.*, Линейное программирование (теория, методы и приложения), М., изд-во «Наука», 1969.
46. *Оуэн Г.*, Теория игр, М., изд-во «Мир», 1971.

Приложения линейного программирования

Металлургия

Fabian T., Application of Linear Programming to Steel Producti Planning, J. Oper. Res. Soc. Am., **3**, 4 (November 1955).
Fabian T., Blast Furnace Burdening and Production Planning, Mana Sci., **14**, 2 (October 1967).
Reinfeid N. V., Do You Want Production or Profit?, Tool. a. Pr duct., **20**, 5 (August 1954).

Угольная промышленность

Henderson J. M., A Short-run Model for the Coal Industry, Rev. Ec nom. a. Statist., **37**, 4 (November 1955).

Химическая промышленность

Arnoff E. L., The Application of Linear Programming, Proceedin of Conference on Operations Research in Production and Inve tory Control, Cliveland, Case Inst. Technol., 1954.
Dantzig G. B., Johnson S., White W., A Linear Programming Appr ach to the Chemical Equilibrium Problem, Manag. Sci., **5**, 1 (C tober 1958).

Нефтяная промышленность

Aronofsky J. S., Williams A. C., The Use of Linear Programming a Mathematics Models in Underground Oil Production, Manag. Sc **8, 4** (July 1962).
Catchpole A. R., The Application of Linear Programming to Integr ted Supply Problems in the Oil Industry, Oper. Res. Qua (U. K.), **13**, 2 (June 1962).
Charnes A., Cooper W. W., Mellon R., Blending Aviation Gasolines A Study in Programmin Interdependent Activities in an Integrat Oil Company, Econometrica, **20**, 2 (April 1952).
Faur P., Elements for Selection of Investments in the Refining I dustry, Proceedings of the Second International Conference Operational Research, N. Y., Wiley, 1960.
Garvin W. W., Crandell H. W., John J. B., Spellman R. A., Applic tions of Linear Programming in the Oil Industry, Manag. Sci., 4 (July 1957).
Manne A. S., Concave Programming For Gasoline Blends, J. Op Res. Soc. Am., **1**, 3 (May 1953).
Manne A. S., Scheduling of Petroleum Refinery Operations, Harva Economic Studies 48, Cambridge, Harvard Univ. Press., 1956.
Symonds G. H., A Crude Allication Problem, см. в книге: *A. S. Ma ne*, Linear Programming: The Solution of Refinery Problems, N. Esso Stand. Oil Comp., 1955.
Symonds C. H., Optimum Production Rates and Inventory to Me Uncertain Seasonal Requirments, см. в книге: *A. S. Manne*, L near Programming: The Solution of Refinery Problems, N. Esso Stand. Oil Comp., 1955.

Vazsonyi A., Optimizing a Function of Additively Separated Variables Subject to a Simple Restriction, Proceedings of the Second Symposium in Linear Programming, **11** (1955).

Бумажная промышленность

Gilmore P. C., Gomory R. E., A Linear-programming Approach to the Cutting-stock Problem — Part I, Oper. Res., **9**, 6 (November — December 1961).
Gilmore P. C., Gomory R. E., A Linear-programming Approach to the Cutting-stock Problem — Part II, Oper. Res., **11**, 6 (November — December 1963).
Morgan J. I., Survey of Operations Research, Paper Mill News, **84**, 11 (March 1961).
Paull A. E., Linear Programming, A Key to Optimum Newsprint Production, Pulp a. Paper Mag. Can., **57**, 1 (January 1956).

Прочие отрасли промышленности

Androit J., Gaussens J., Programme of Thermal Reactor and «Breeder Reactor» Power Stations — Fuel Economy Problem of Storage — Price of Plutonium, Proceedings of the Second International Conference on Operational Research, N. Y. Wiley, 1960.
Charnes A., Cooper W. W., Ferguson R. O., Optimal Estimation of Executive Compensation by Linear Programming, Manag. Sci., **1**, 2 (January 1955).
Horowitz J., Lattes R., Parker E., Some Operational Problems Connected with Power Reactor Discharges, Proceedings of the Second International Conference on Operational Research, N. Y., Wiley, 1960.

Транспорт и средства связи

Charnes A., Miller M. H., A Model for Optimal Programming of Railway Freight Train Movements, Manag. Sci. **3**, 1 (October 1956).
Crane R. R., A New Tool: Operations Research, Mod. Railroads, **9**, 1 (January 1954).
Morton G., Application of Linear Programming Methods to Commercial Airline Operations, Econometrica, **21**, 1 (January 1953).
Kalaba R. E., Juncosa M. L., Optical Design and Utilization of Communication Networks, Manag. Sci., **3**, 1 (October 1956).
Saaty T. L., Suzuki G., A Nonlinear Programming Model in Optimum Communication Satellite Use, SIAM Rev., **7**, 3 (July 1965).

Планирование производства и хранение продукции

Bellman R. E., Mathematical Aspects of Scheduling Theory, J. Ind. a. Appl. Math., **4**, 3 (September 1956).
Cahn A. S., Jr., The Warehouse Problem (Abstract 505), Bull. Am. Math. Soc., **54** (November 1948).
Charnes A., Cooper W. W., Generalizations of the Warehousing Model, Oper. Res. Quart., **6**, 4 (December 1955).

Charnes A., Cooper W. W., Farr D., Linear Programming and Profit Preference Scheduling for a Manufacturing Firm, J. Oper. Res. Soc. Am., **1**, 3 (May 1953).
Charnes A., Cooper W. W., Mellon B., A Model for Optimizing Production by Reference to Cost Surrogates, Econometrica, **23**, 3 (July 1955).
Efroymson M. A., Ray T. L., A Branch-bound Algorithm for Plant Location, Oper. Res., **14**, 3 (May — June 1966).
Fetter R. B., A Linear Programming Model for Long Range Capacity Planning, Manag. Sci., **7**, 4 (July 1966).
Gomory R. E., Dzielinski B. P., Optimal Programming of Lot Sizes, Inventory and Labor Allocation, Manag. Sci., **11**, 9 (July 1965).
Gepfert A., Grace Ch. H., Operations Research... as It is Applied to Production Problems, Tool Eng., **36**, 5 (May 1956).
Johnson S. M., Optimal Two-and Three-stage Production Schedules with Setup Times Included, Nav. Res. Logist. Quart., **1**, 1 (March 1954).
Канторович Л. В., Методы оптимизации и математические модели экономики. УМН, **25**, 5, 1970.
Kelley J. E., Jr., Critical Path Planning and Scheduling: Mathematical Basis, Oper. Res., **9**, 3 (May — June 1961).
Koeningsberg E., Some Industrial Applications of Linear Programming, Oper. Res. Quart. (U. K.), **19**, 2 (June 1961).
Magee J. R., Guides To Inventory Policy, Part I: Functions and Lot Sizes, Harv. Bus. Rev., **34**, 1 (January — February 1956); Part II: Problems of Uncertainty, там же, **34**, 2 (March — April 1956); Part III: Anticipating Futur Needs, там же, **34**, 3 (May — June 1956).
Magee J. R., Linear Programming in Production Scheduling, J. Oper. Res. Soc. Am., **1**, 2 (February 1953).
Manne A. S., An Application of Linear Programming to the Procurement of Transport Aircraft, Manag. Sci., **2**, 2 (January 1956).
Rapoport L. A., Drews W. P., Mathematical Approach to Long-range Planning, Harvard Business Rev., **40**, 3 (May—June 1962).
Salveson M. E., The Assembly Line Balancing Problem, Proceeding of the Second Symposium in Linear Programming, **1**, p. 55 (1955).
Smith S. B, Planning Transistor Production by Linear Programming, Oper. Res., **13**, 1 (January—February 1965).
Vazsonyi A., A Problem in Machine Shop Loading, J. Oper. Res. Sos. Am., **3**, 1 (February 1955).
Whitin T. M., Inventory Control Research: A Survey, Manag. Sci., **1**, 1 (October 1954).

Сельское хозяйство

Boles J. N., Linear Programming and Farm Management Analysis, J. Farm Econom., **37**, 1 (February 1955).
Candler W., A Modified Simplex Solution for Linear Programming with Variable Capital Restriction, J. Farm Econom., **38**, 4 (November 1956).
Fisher W. D., Shruben L. W., Linear Programming Applied to Feedmixing under Different price Conditions, J. Farm Econom., **35**, 4 (November 1953).
Fox K. A., Taeuber R. C., Spatial Equilibrium Models of the Livestock-feed Economy, Am. Econom. Rev., **45**, 4 (September 1955).

Fox K. A., Taeuber R. C., A Spatial Equilibrium Model of the Livestock-feed Economy in the United States, Econometrica, **21**, 4 (October 1953).
Hildreth C. G., Economic Implications of Some Cotton Fertilizer Experiments, Econometrica, **23**, 1 (January 1955).
Hildreth C. G., Reiter S., On the Choice of a Crop Rotation Plan, см. в книге [27].
King P. A., Use of Economic Models: Some Applications of Activity Analysis in Agricultural Economics, J. Farm Econom., **35**, 5 (December 1953).
Pierson D. R., Farm Profits Up by 40 Percent, Authomat. Data Proces., **4**, 5 (May 1962).
Swanson E. R., Integrating Crop and Livestock Activities in Farm Management Activity Analysis, J. Farm Econom., **37**, 5 (December 1955).
Swanson L. W., Woodruff J. G., A Sequential Approach to the Feedmix Problem, Oper. Res., **12**, 1 (January — February 1964).

Экономические исследования

Altshul E., Reorientation in Economic Theory: Linear and Nonlinear Programming, Am. Math. Month., **62**, 6 (June 1955).
Beckmann M. J., Marschak T., An Activity Analysis Approach to Location Theory, Proceedings of the Second Symposium in Linear Programming, **I**, p. 331, 1955.
Brown J. A. C., An Experiment in Demand Analysis: The Computation of the Manchester Computer, Oper. Res. (JORSA), **4**, 1 (February 1956).
Charnes A., Cooper W. W., An Example of Constrained Games in Industrial Economics, Econometrica, **22**, 4 (October 1954).
Chipmann J., Linear Programming, Rev. Econom. a. Statist., **35**, 2 (May 1953).
Davidson D., Suppes P., Experimental Measurement of Utility by Use of a Linear Programming Model, Econometrica, **24**, 2 (April 1956).
Dorfman R., Application of Linear Programming to the Theory of the Firm, Including and Analysis of Monopolistic Firms by Nonlinear Programming, Econometrica, **22**, 1 (January 1954).
Gunter P., Use of Linear Programming in Capital Budgeting, J. Oper. Res. Soc. Am., **3**, 2 (May 1955).
Markowitz H. M., Portfolio Selection, J. Finance, **7**, 1 (March 1952).
Martin A. D., Jr., Mathematical Programming of Portfolio Selections, Manag. Sci., **1**, 2 (January 1955).
Samuelson P. A., Linear Programming and Economic Theory, Proceedings of the Second Symposium in Linear Programming, **I**, p. 251, 1955.
Whitin Th. M., Classical Theory, Graham's Theory and Linear Programming in International Trade, Quart. J. Econom., **67**, 4 (November 1953).

Расчеты и конструирование

Charnes A., Greenberg H. J., Plastic Collapse and Linear Programming, Preliminary Report, Bull. Soc., **57**, 6 (November 1951).
Dorn W. S., Greenberg H. J., Linear Programming and Plastic Limit Analysis of Structures, Technical Report 7, Pittsburgh, Carnegie Inst. Techn. (August 1955).

Heyman J., Plastic Design of Beams and Plane Frames for Minimum Material Consumption, Quart. Appl. Math., **8**, 4 (January 1951).

Регулирование уличного движения

Lavallee R. S., The Application of Linear Programming to the Problem of Scheduling Traffic Signals, J. Oper. Res. Soc. Am., **3**, 4 (November 1955).
Little J. P. C., The Synchronization of Traffic Signals by Mixedinteger Linear Programming, Oper. Res., **14**, 4 (July — August 1966).

Прочие приложения

Cohen K. J., Hammer F. S., Optimal Coupon Schedules for Municipal Bonds. Manag. Sci., **12**, 1 (September 1965).
Ford L., Fulkerson D. R., Maximal Flow through a Network, Can. J. Math., **8**, 3 (1956).
Freeman R. J., Gogerty D. C., Graves G. W., Brooks R. B. S., A Mathematical Model of Supply Support for Space Operations, Oper. Res., **14**, 1 (January — February 1966).
Hartung P. H., Brand Switching and Mathematical Programming in Market Expansion, Manag. Sci., **11**, 10 (August 1965).
Knight U. G. W., The Logical Design of Electrical Networks Using Linear Programming Methodes, Inst. Electr. Eng. Proceed., **107**, 33 (June 1960).
Kolesar P. J., Linear Programming and the Reliability of Multicomponent Systems, Nav. Res. Logist. Quart., **14**, 3 (September 1967).
Loucks D. P., Revelle C. S., Lynn W. R., Linear Programming Models for Water Pollution Control, Manag. Sci., **4** (December 1967).
McGuire C. B., Some Team Models of a Sales Organisation, Manag. Sci., **7**, 2 (January 1961).
Mannos M., An Application of Linear Programming to Efficiency in Operations of a System of Dams, Econometrica, **23**, 3 (July 1955).
Orden A., Application of Linear Programming to Optical Filter Design, Proceedings of the Second Symposium in Linear Programming, **I**, p. 185 (January 1955).
Wagner H. M., Giglio R. S., Glaser R. G., Preventive Maintenance Scheduling by Mathematical Programming, Manag. Sci., **10**, 2 (January 1964).
Wardle P. A., Forest Management and Operational Research: A Linear Programming Study, Manag. Sci., **11**, 10 (August 1965).
Wilson R. C., A Packaging Problem, Manag. Sci., **12**, 4 (December 1965).
Percus J., Quinto L., The Application of Linear Programming to Competitive Bond Bidding, Econometrica, **24**, 4 (October 1956).
Stanley E. D., Honig D., Gainen L., Linear Programming in Bid Evaluation. Nav. Res. Logist. Quart., **1**, 1 (1954).
Waggener H. A., Suzuki G., Bid Evaluation for Procurement of Aviation Fuel at DFSC: A Case History, Nav. Res. Logist. Quart., **14**, 1 (March 1967).
Jacobs W. W, The Caterer Problem, Nav. Res. Logist. Quart., **1**, 2 (June 1953).

Jacobs W. W., Military Applications of Linear Programming, Proceedings of the Second Symposium in Linear Programming, **I**, p. 1 (1955).
Nicholson G. E., Blackwell G. W., Game Theory and Defence against Community Disaster, Res. Prev., **2**, 3 (May 1954).
Saaty T. L., Webb K. W., Sensitivity and Renewals in Scheduling Aircraft Overhaul, Proceedings of the Second International Conference on Operational Research, N. Y., Wiley, 1960.
Wood M. K., Geisler M. A., Development of Dynamic Models for Programm Planning, см. в книге: Activity Analysis of Production and Allocation, Koopmance, ed., N. Y., Wiley, 1951.

Транспортная задача и теория сетей

Balinski M. L., Quandt R. E., On an Integer Program for a Delivery Problem, Oper. Res., **12**, 2 (March — April 1964).
Dantzig G. B., Johnson D. L., Maximum Payloads per Unit Time Delivered through an Air Network, Oper. Res., **12**, 2 (March — April, 1964).
Dantzig G. B., Ramser J. H., The Truck Dispatching Problem, Manag. Sci., **6**, 1 (October 1959).
Dwyer P. S., The Solution of the Hitchcock Transportation Problem with a Method of Reduced Matrices, Ann. Arbor, Univ. Michigan, December 1955.
Flood M. M., Application of Transportation Theory to Scheduling a Military Tanker Fleet, J. Oper. Res. Soc. Am., **2**, 2 (May 1954).
Flood M. M., On the Hitchcock Distribution Problem, Pacific J. Math., **3**, 2 (June (1953).
Fulkerson D. R., Dantzig G. B., Computation of Maximal Flows in Networks, Nav. Res. Logist. Quart., 2, 4 (December 1955).
Gleyzel A. N., An Algorithm for Solving the Transportation Problem (Research Paper 2583). J. Res. Nat. Bureau Stand., **54**, 4 (April 1955).
Hitchcock F. L., The Distribution of a Product from Several Sources to Numerous Localities, J. Math. Physics, **20**, 3, 224 (August 1941).
Канторович Л. В., О перемещении масс, ДАН СССР, **37**, 7—8, 1942.
Koopmans T. C., Optimum Utilization of the Transportation System, Econometrica, **17**, 3 and 4 (July 1949).
Lederman J., Gleiberman L., Egan J. F., Vessel Allocation by Linear Programming, Nav. Res. Logist. Quart., **13**, 3 (September 1966).
Orden A., The Transshipment Problem, Manag. Sci., **2**, 2 (April 1956).

Задача коммивояжера

Dantzig G. B., Fulkerson D. R., Johnson S., Solution of a Largescale Travelling-salesman Problem, J. Oper. Res. Soc. Am., **2**, 4 (November 1954).
Flood M. N., The Travelling-salesman Problem, Oper. Res., **4**, 1 (February 1956).
Heller I., On the Travelling Salesman's Problem, Proceedings of the Second Symposium in Linear Programming, **II**, p. 643 (January 1955).
Little J. D. C., Murty K. G., Sweeney D. W., Karel C., An Algorithm for the Travelling-salesman Problem, Oper. Res., **11**, 6 (November — December 1963).

СОДЕРЖАНИЕ

С. Гасс

ПУТЕШЕСТВИЕ В СТРАНУ ЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ

Редактор А. Г. Белевцева
Художественный редактор Ю. Л. Максимов
Технический редактор Л. П. Бирюкова
Корректор В. М. Дюжева

Сдано в набор 25/IX 1972 г. Подписано к печати 11/XII 1972 г. Бумага № 1 $84\times108^{1}/_{32}=2{,}75$ бум. л. 9,24 усл. печ. л. Уч.-изд. л. 8,67. Изд. № 12/6521
Цена 44 коп. Зак. 327

ИЗДАТЕЛЬСТВО «МИР»
Москва, 1-й Рижский пер., 2

Ордена Трудового Красного Знамени Ленинградская типография № 2
имени Евгении Соколовой «Союзполиграфпрома»
при Государственном комитете Совета Министров СССР по делам
издательств, полиграфии и книжной торговли
г. Ленинград, Л-52, Измайловский проспект, 29

ИЗДАТЕЛЬСТВО МИР

44 к.