МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования
«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
(ФГБОУ ВПО «КубГУ»)

Кафедра прикладной математики

КУРСОВАЯ РАБОТА

МАТЕМАТИЧЕСКАЯ МОДЕЛЬ ПОСТАВКИ И ХРАНЕНИЯ ПРОДУКЦИИ

Работу выполнила _____ С.А. Глинкова
(подпись, дата) (инициалы, фамилия)

Факультет компьютерных технологий и прикладной математики 3 курс

Направление 01.03.02 – Прикладная математика и информатика

Научный руководитель,
доц., канд. физ.-мат. наук _____ А.Д. Колотий
(подпись, дата) (инициалы, фамилия)

Нормоконтролер,
магистр педагогики _____ О.Н.Свистунова
(подпись, дата) (инициалы, фамилия)

Краснодар 2015

СОДЕРЖАНИЕ

Введение .....**Ошибка! Закладка не определена.**
1 Понятие моделирования 4
1.1 Моделирование 4
1.2 Математическое моделирование 4
2 Дискретное программирование и его понятия 8
2.1 Целочисленное линейное программирование 8
2.2 Метод ветвей и границ 10
2.3 Поиск решения в MS Office Excel 13
3 Задача планирования производства .....**Ошибка! Закладка не определена.**
3.1 Первая модель .....**Ошибка! Закладка не определена.**
3.2 Вторая модель .....**Ошибка! Закладка не определена.**
4 Примеры задач 20
4.1 Задача о комивояжёре 20
4.2 Задача календарного планирования трех станков 24
Список использованных источников.....28

ВВЕДЕНИЕ

В настоящее время вопросы хранения и поставки продукции принимают особо важное экономическое значение, особенно это касается продовольственной продукции. Каждый из товаров нуждается при хранении в определенном режиме, зависящем от состава, свойств и протекающих в нем процессов. На поиск нахождения оптимальных условий хранения продукции и кратчайших путей поставки этой же продукции в пункты назначения может уйти много времени. К счастью, подобная проблема может решаться достаточно легко средствами математического моделирования. Решение будет найдено с достаточной точностью, с малыми затратами, что является преимуществом подобного пути. Зная эти методы, и умея применять их на практике, в дальнейшем возможно ставить и реализовывать наиболее глобальные по времени и стоимости задачи хранения и поставки продукции. Это существенно облегчит развитие производства без лишних затрат.

1 Понятия моделирования

1.1 Моделирование

Модель – это материальный или мысленно представленный объект, который в процессе познания(изучения) замещает оригинал, сохраняя некоторые важные для данного исследования типичные свойства [9].

Модель в широком смысле – любой образ, аналог мысленный или установленный: изображение, описание, схема, чертеж, карта и т.п. какого-либо объема, процесса или явления, используемый в качестве его заменителя или представителя. Сам объект, процесс или явление называется оригиналом данной модели.

Моделирование – это исследование какого-либо объекта или системы объектов путем построения и изучения их моделей. Это использование моделей для определения или уточнения характеристик и рационализации способов построения вновь конструируемых объектов.

1.2 Математическое моделирование

Математическая модель – приближенное описание объекта моделирования, выраженное с помощью математической символики [7].

Процесс построения и изучения математических моделей называется математическим моделированием.

Реализованная на компьютере математическая модель называется компьютерной математической моделью, а проведение целенаправленных расчетов с помощью компьютерной модели называется вычислительным экспериментом.

Этапы компьютерного математического моделирования:

Первый этап — определение целей моделирования. Эти цели могут быть различными:

1 Модель нужна для того, чтобы понять, как устроен конкретный объект, какова его структура, основные свойства, законы развития и взаимодействия с окружающим миром (понимание);

2 Модель нужна для того, чтобы научиться управлять объектом (или процессом) и определить наилучшие способы управления при заданных целях и критериях (управление);

3 Модель нужна для того, чтобы прогнозировать прямые и косвенные последствия реализации заданных способов и форм воздействия на объект (прогнозирование).

Второй этап: определение входных и выходных параметров модели; разделение входных параметров по степени важности влияния их изменений на выходные. Такой процесс называется ранжированием, или разделением по рангам.

Третий этап: построение математической модели. На этом этапе происходит переход от абстрактной формулировки модели к формулировке, имеющей конкретное математическое представление.

Четвертый этап: выбор метода исследования математической модели. Чаще всего здесь используются численные методы, которые хорошо поддаются программированию. Как правило, для решения одной и той же задачи подходит несколько методов, различающихся точностью, устойчивостью и т.д. От верного выбора метода часто зависит успех всего процесса моделирования.

Пятый этап: разработка алгоритма, составление и отладка программы для ЭВМ — трудно формализуемый процесс.

Шестой этап: тестирование программы. Работа программы проверяется на тестовой задаче с заранее известным ответом. Это — лишь начало процедуры тестирования, которую трудно описать формально исчерпывающим образом. Обычно тестирование заканчивается тогда, когда пользователь по своим профессиональным признакам сочтет программу верной.

Седьмой этап: собственно, вычислительный эксперимент, в процессе которого выясняется, соответствует ли модель реальному объекту (процессу).

Модель достаточно адекватна реальному процессу, если некоторые характеристики процесса, полученные на ЭВМ, совпадают с экспериментально полученными характеристиками с заданной степенью точности. В случае несоответствия модели реальному процессу возвращаемся к одному из предыдущих этапов.

Классификация математических моделей:

1 Детерминированные модели - это модели, в которых установлено взаимно-однозначное соответствие между переменными описывающими объект или явления.

2 Стохастические модели – это модели, связь между переменными в которых носит случайный характер. Иногда это бывает принципиально. Воздействие огромного количества факторов, их сочетание приводит к случайному набору переменных описывающих объект или явление.

3 Статистическая модель – это модель, включающая описание связей между основными переменными моделируемого объекта в установившемся режиме без учета изменения параметров во времени.

4 Динамическая модель – это модель, в которой описываются связи между основными переменными моделируемого объекта при переходе от одного режима к другому.

5 Непрерывная модель – это модель, в которой переменные принимают значения из некоторого промежутка.

6 Дискретная модель – это модель, в которой переменные принимают изолированные значения.

7 Линейная модель – это модель, в которой все функции и отношения, описывающие модель, линейно зависят от переменных и не линейная в противном случае.

Требования, предъявляемые к моделям:

1 Универсальность – характеризует полноту отображения моделью изучаемых свойств реального объекта.

2 Адекватность – способность отражать нужные свойства объекта с погрешностью не выше заданной.

3 Точность – оценивается степенью совпадения значений характеристик реального объекта и значения этих характеристик полученных с помощью моделей.

4 Экономичность – определяется затратами ресурсов ЭВМ памяти и времени на ее реализацию и эксплуатацию.

2 Дискретное программирование и его понятия

Дискретное программирование – область математики, занимающаяся исследованием и решением экстремальных задач на конечных множествах.
В экономике существует огромное количество задач с дискретной природой. В первую очередь это задачи с физической неделимостью многих факторов и объектов расчета. Дискретными являются задачи с логическими переменными, принимающими только два значения: 0 и 1 (да и нет).

Наиболее распространенными задачами дискретного программирования являются задачи:

1 О контейнерных перевозках, в которой определяется максимум перевезенной продукции при ограничениях на вместимость;

2 О назначениях, в которой определяется наиболее рациональное использование потенциала сотрудников при выполнении определенных работ;

3 Коммивояжера, в которой определяется минимальный путь агента, проходящий через необходимые пункты возможного сбыта продукции;

4 Транспортная, один из случаев задачи дискретного программирования.

Из всего множества задач, определенных на конечных множествах, дискретное программирование занимается только нетривиальными задачами, то есть теми, которые не могут быть решены простым просмотром.

Трудности, возникающие при решении задач дискретного программирования, связаны, в первую очередь, с большим числом локальных экстремумов и с тем, что множество допустимых значений задано неявно.

Наиболее изученными задачами дискретного программирования являются целочисленные задачи линейного программирования.

2.1 Целочисленное линейное программирование

Под задачей целочисленного линейного программирования (ЦЛП) понимается задача линейного программирования, в которой некоторые (а возможно, и все) переменные должны принимать целые значения. Задача

целочисленного линейного программирования называется полностью целочисленной, если все её переменные должны быть целочисленными. Для смешанной задачи целочисленного линейного программирования лишь некоторые переменные предполагаются целочисленными, а остальные могут принимать произвольные (нецелые) значения [6].

Математическая модель следующего вида называется моделью смешанного целочисленного линейного программирования, записанная в стандартной форме (сокращенно *MIP*, от англ. mixed integer programming):

$\min(cx + fy)$

$Ax + By \geq b$,

$x \geq 0$,

$y \geq 0$, целые,

где $c = (c_1,\ldots,c_n)$, $f = (f_1,\ldots,f_p)$ — вектор - строки с вещественными компонентами, задающие коэффициенты целевой функции;

$x = (x_1, \ldots, x_n)^T$ — вектор - столбец переменных задачи, принимающих неотрицательные вещественные значения;

$y = (y_1, \ldots, y_p)^T$ — вектор - столбец переменных задачи, принимающих неотрицательные целочисленные значения;

A, B – матрицы размерности $(m \times n)$ и $(m \times p)$ соответственно, с рациональными

значениями компонент, определяющие коэффициенты в системе из m ограничений;

$b = (b_1, \ldots, b_m)^T$ — вектор - столбец с вещественными компонентами правых частей ограничений.

В стандартной форме требуется, чтобы задача была на минимум, все знаки в ограничениях были нестрогими, вида ≥, а переменные принимали неотрицательные значения.

Существуют каноническая и общая формы записи *MIP*. Каждую из этих форм можно преобразовать в стандартную, пользуясь следующими простыми соображениями:

— ограничения равенства $Ax + By = b$ эквивалентны паре неравенств $Ax + By \geq b$ и $-Ax - By \geq -b$;

— задача на максимум с целевой функцией $(cx + fy)$ эквивалентна задаче на минимум с целевой функцией $-(cx + fy)$: аналогичные преобразования, т. е. умножение на (-1), нужно сделать в неравенствах со знаком $\leq$;

— каждое вхождение свободной переменной x, т. е. переменной без ограничения на знак, нужно заменить разностью двух новых неотрицательных переменных, $x = (u - v)$. Таким образом, все переменные в модели будут неотрицательными.

Любое оптимальное решение, полученное для модели в стандартной форме, нетрудно преобразовать в оптимальное решение исходной задачи, и наоборот.

Задача *MIP*, в которой все переменные принимают значения только 0 или 1, называют задачей булева, или 0–1, программирования.

Задача *MIP*, в которой все целочисленные переменные принимают значения 0 или 1, называют задачей смешанного 0–1 программирования.

Задача *MIP*, в которой все переменные принимают вещественные значения, называют задачей линейного программирования (сокращенно *LP*, от англ. linear programming).

Задача *MIP*, в которой все переменные принимают только целые значения, называют задачей полностью целочисленного программирования (сокращенно *IP*, от англ. integer programming).

Одним из методов решения как полностью целочисленных, так и смешанных задач ЦЛП является метод ветвей и границ. Он представляет собой эффективную процедуру перебора всех целочисленных допустимых решений.

2.2 Метод ветвей и границ

Метод ветвей и границ впервые предложили в 1960 году Ленд и Дойг для решения задач целочисленного программирования.

По существу, метод является вариацией полного перебора с отсевом подмножеств допустимых решений, заведомо не содержащих оптимальных решений.

Описание метода:

В основе метода ветвей и границ лежит идея последовательного разбиения множества допустимых решений на подмножества. На каждом шаге метода элементы разбиения подвергаются проверке для выяснения, содержит данное подмножество оптимальное решение или нет. Проверка осуществляется посредством вычисления оценки снизу для целевой функции на данном подмножестве. Если оценка снизу не меньше рекорда – наилучшего из найденных решений, то подмножество может быть отброшено. Проверяемое подмножество может быть отброшено еще и в том случае, когда в нем удается найти наилучшее решение. Если значение целевой функции на найденном решении меньше рекорда, то происходит смена рекорда. По окончанию работы алгоритма рекорд является результатом его работы [2].

Если удается отбросить все элементы разбиения, то рекорд – оптимальное решение задачи. В противном случае, из не отброшенных подмножеств выбирается наиболее перспективное (например, с наименьшим значением нижней оценки), и оно подвергается разбиению. Новые подмножества вновь подвергаются проверке и т.д.

При применении метода ветвей и границ к каждой конкретной задаче в первую очередь должны быть определены две важнейшие его процедуры:

1 Ветвления множества возможных решений;

2 Вычисления нижних и верхних оценок целевой функции.

Правила ветвления:

В зависимости от особенностей задачи для организации ветвления обычно используется один из двух способов:

1 Ветвление множества допустимых решений исходной задачи D;

2 Ветвление множества D', получаемого из D путем снятия условия целочисленности на переменные.

Первый способ ветвления обычно применяется для задач целочисленного программирования и заключается в выделении подобластей возможных решений путем фиксации значений отдельных компонент целочисленных оптимизационных переменных.

Второй способ ветвления - более универсальный, чем первый. Для осуществления ветвления некоторой области D_i' этим способом на D_i' решается оптимизационная задача с целевой функцией исходной задачи и действительными переменными.

Ветвление осуществляется, если в оптимальном решении значение хотя бы одной целочисленной по исходной постановке задача переменной не является целочисленным. Среди этих переменных выбирается одна, например, j - я. Обозначим ее значение в найденном оптимальном решении x^0[j]. Говорят, что ветвление осуществляется по переменной x[j]. Область D_i' разделяется на две подобласти D_{i1}' и D_{i2}' следующим образом:

$$D_{i1}' = D_i' \cap (x[j] \le [x^0[j]])$$

$$D_{i2}' = D_i' \cap (x[j] \ge [x^0[j]] + 1),$$

где $[x^0[j]]$ – целая часть значения $x^0[j]$.

Алгоритм метода ветвей и границ:

Основные правила алгоритма могут быть сформулированы следующим образом:

1 Ветвлению в первую очередь подвергается подмножество с номером s(s ≤ 1), которому соответствует наименьшее значение нижней оценки целевой функции (I - это множество номеров всех подмножеств, D_i (или D_i'), находящихся на концах ветвей и ветвление которых еще не прекращено). Если реализуется изложенный выше способ ветвления множеств D_i', то может возникнуть неоднозначность относительно выбора компоненты, по которой необходимо осуществлять очередной шаг ветвления [3]. К сожалению, вопрос о «наилучшем» способе такого выбора с общих позиций пока не решен, и поэтому в конкретных задачах используются некоторые эвристические правила.

2 Если для некоторого i-го подмножества выполняется условие $\varepsilon_i \geq \mu(D)$, то ветвление его необходимо прекратить, так как потенциальные возможности нахождения хорошего решения в этом подмножестве (их характеризует ε_i) оказываются хуже, чем значение целевой функции для реального, найденного к данному моменту времени, допустимого решения исходной задачи (оно характеризует μ(D)).

3 Ветвление подмножества D_i' прекращается, если найдено в задаче оптимальное решение $x_D^0 \in D$. Обосновывается это тем, что $f\left(x_{D_i}^0\right) = \varepsilon_i$, и, следовательно, лучшего допустимого решения, чем $x_{D_i'}^0$ в этом подмножестве не существует. В этом случае рассматривается возможность корректировки μ(D).

4 Если $\varepsilon \geq \mu(D)$, где $\varepsilon = min\varepsilon_i$, то выполняются условия оптимальности для найденного к этому моменту лучшего допустимого решения [4]. Обоснование такое же, как и пункта 2 настоящих правил.

5 После нахождения хотя бы одного допустимого решения исходной задачи может быть рассмотрена возможность остановки работы алгоритма с оценкой Δ близости лучшего из полученных допустимых решений к оптимальному (по значению целевой функции):

$$\Delta = \mu(D) - \varepsilon$$

2.3 Поиск решения в MS Office Excel

Поиск решений – надстройка Excel, которая помогает найти решение с помощью изменения значений целевых ячеек. Целью может быть минимизация, максимизация или достижение некоторого целевого значения. Проблема решается путем регулировки входных критериев или ограничений, определенных пользователем.

Где в Excel поиск решений:

Надстройка Поиск решений поставляется вместе с Excel, но по умолчанию отключена. Чтобы включить его, перейдите по вкладке Файл в группу Параметры**.** В появившемся диалоговом окне Параметры,

выберите Надстройки → Управление: Надстройки Excel → Перейти. В окне Надстройки устанавливаем галочку напротив поля Поиск решения, (Рисунок 1), жмем ОК.

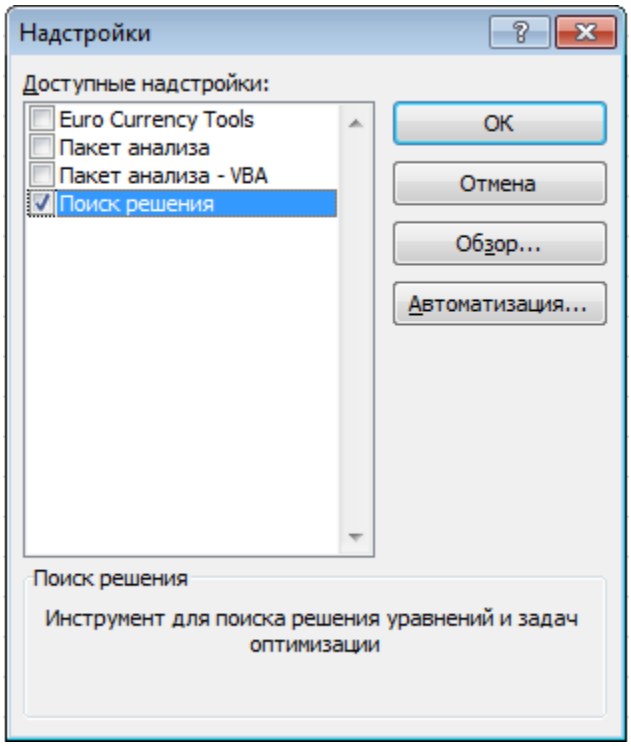


Рисунок 1 – Надстройки Excel

Теперь во вкладке Данные появилась новая группа Анализ с кнопкой Поиск решения (Рисунок 2).

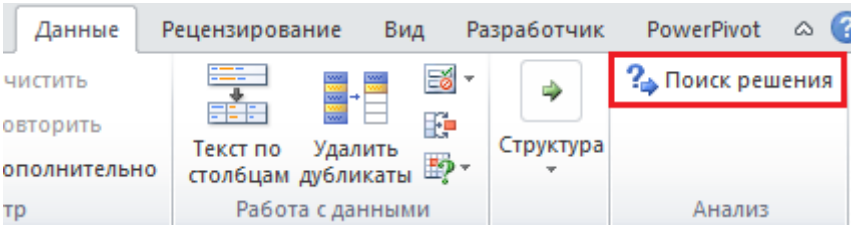


Рисунок 2 – Кнопка Поиск решения

3 Задача планирования производства

Рассмотрим математическую модель поставки и хранения продукции на примере задачи планирования производства.

Постановка задачи:

Завод занимается производством некоторой однотипной продукции. Продукция производится в начале месяца и либо отправляется потребителям сразу, либо хранится на складе. Продукцию можно производить и хранить на складе на протяжении всего горизонта планирования. Производственные затраты складываются из единовременных расходов на запуск производства, удельных производственных затрат и удельных затрат на хранение продукции. Задан промежуток времени T месяцев, в течение которого необходимо спланировать производство и хранение продукции так, чтобы выполнить заказ с минимальными затратами [1].

Для данной задачи обозначим через d_t — заказ на продукцию в месяц t, c_t — затраты на запуск производства в месяц t, p_t — удельные затраты на производство в месяц t, h_t — удельные затраты на хранение продукции в течение месяца t. Эти значения являются известными.

Данная задача решается при помощи математической модели смешанного целочисленного линейного программирования. Для её решения можно использовать две модели с разными переменными.

3.1 Первая модель

Первая модель использует следующие переменные:

y_t — количество продукции, произведенной в месяц t;

s_t — количество продукции, оставленной на складе к концу месяца t;

$$x_t = \begin{cases} 1, & \text{если в месяц } t \text{ было запущено производство,} \\ 0 - & \text{в противном случае.} \end{cases}$$

Тогда получаем следующую математическую модель:

$$min \sum_{t=1}^{T} (c_t x_t + p_t y_t + h_t s_t) \quad (1)$$

$$y_1 = d_1 + s_1, \quad (2)$$

$$s_{t-1} + y_t = d_t + s_t, t = 2, \dots, T, \quad (3)$$

$$y_t \leq \sum_{k=1}^{T} d_k x_k, \; t = 1, \dots, T, \quad (4)$$

$$s_T = 0, \quad (5)$$

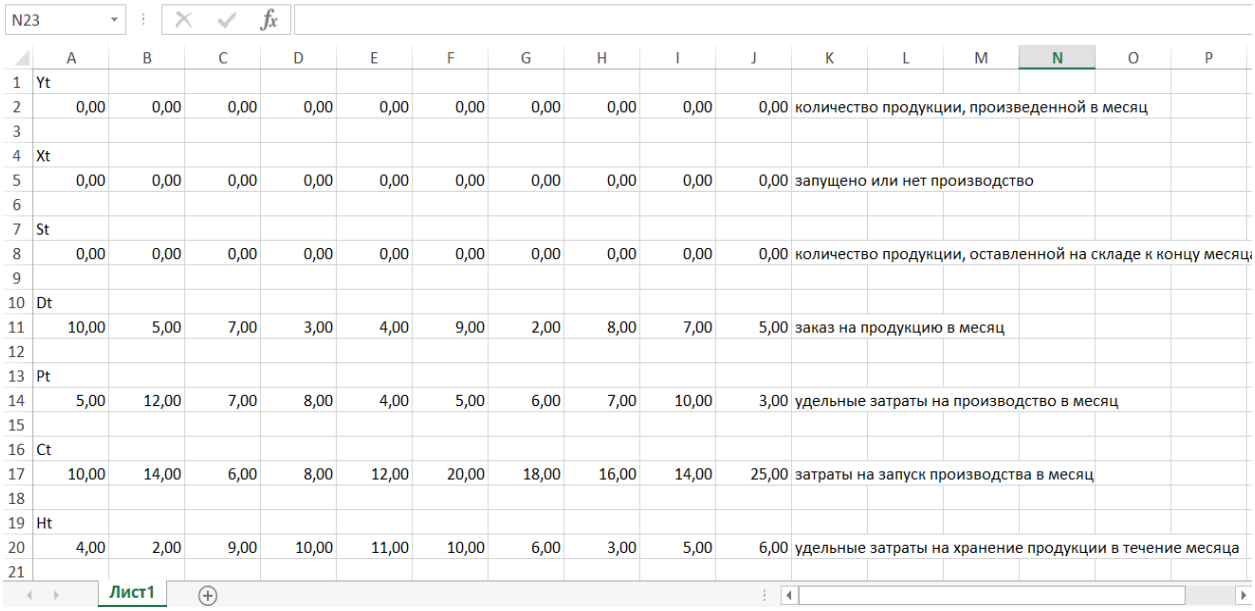
$$y_t \geq 0, s_t \geq 0, \text{целые}, x_t \in \{0,1\}, t = 1, \dots, T. \quad (6)$$

Целевая функция (1) — минимальные общие затраты, которые складываются из затрат на запуск, производство и хранение. Ограничения (2) и (3) отвечают за удовлетворение спроса и выполнение «закона сохранения» продукции в каждом месяце. Поскольку в начальный момент времени на складе отсутствует продукция, то для первого месяца выписано отдельное условие.

Ограничения (4) устанавливают связь между переменными x_t и y_t и ограничивают сверху максимальный ежемесячный объем партии. Только если производство запущено в месяц *t*, на нем может производится продукция, но не более, чем величина суммарного спроса на него. Ограничение (5) задает уровень запасов на складе в конце горизонта планирования.

Рассмотрим данную модель на простейшем примере, реализованном при помощи поиска решения средств MS Office Excel.

На рисунке 3 показаны изначальные данные, оформленные в строковом виде:



	A	B	C	D	E	F	G	H	I	J	K
1	Yt										
2	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	количество продукции, произведенной в месяц
3											
4	Xt										
5	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	запущено или нет производство
6											
7	St										
8	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	количество продукции, оставленной на складе к концу месяц
9											
10	Dt										
11	10,00	5,00	7,00	3,00	4,00	9,00	2,00	8,00	7,00	5,00	заказ на продукцию в месяц
12											
13	Pt										
14	5,00	12,00	7,00	8,00	4,00	5,00	6,00	7,00	10,00	3,00	удельные затраты на производство в месяц
15											
16	Ct										
17	10,00	14,00	6,00	8,00	12,00	20,00	18,00	16,00	14,00	25,00	затраты на запуск производства в месяц
18											
19	Ht										
20	4,00	2,00	9,00	10,00	11,00	10,00	6,00	3,00	5,00	6,00	удельные затраты на хранение продукции в течение месяца
21											

Рисунок 3 – Первоначальные данные задачи

Данные в строках задаются в тысячах рублей.

Для задачи зададим вспомогательные условия. Рассчитаем сумму всех заказов на продукцию для заданных месяцев. Обозначим это сумму буквой S. В данном примере S = 60 тыс. Для нахождения целевой функции (обозначим F), найдем три, изменяемых в процессе, слагаемых и сложим их. Первое слагаемое: затраты на запуск производства в месяц, умноженные на вероятность запуска производства. Второе слагаемое: удельные затраты на производство в месяц, умноженные на количество продукции, произведенной в месяц. Третье слагаемое: удельные затраты на хранение продукции в течение месяца, умноженные на количество продукции, оставленной на складе к концу месяца. Обозначим их соответственно F1, F2, F3. Используя ограничения (2) – (6), изменяя ячейки переменных массивов y_t, s_t, x_t, по целевой функции F, поставим задачу целочисленного линейного программирования при помощи поиска решения (Рисунок 4):

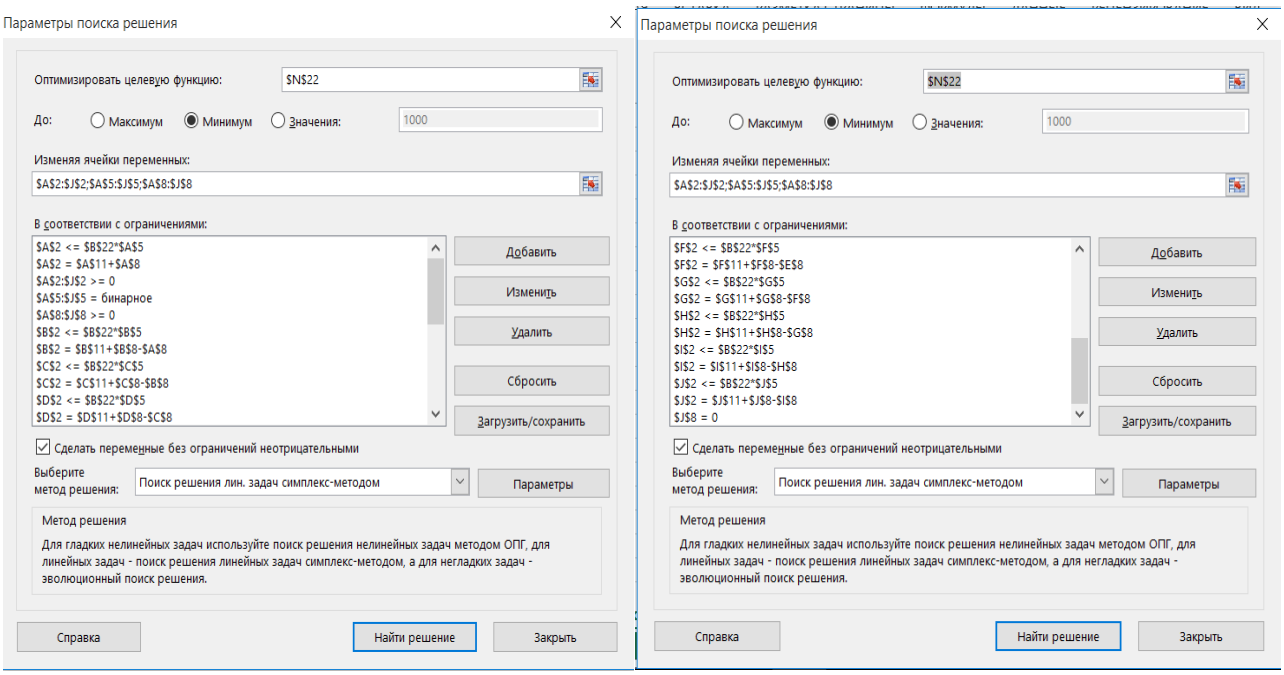


Рисунок 4 – Поиск решения для линейной задачи

В результате работы программы было получено значение минимальных общих затрат на производство данного примера F = 361 тыс., а также рассчитан наиболее оптимальный план, соответствующий этим затратам. Согласно этому плану, производство будет запущенно в 1, 2, 4 – 6, 8, 10 месяцах. Затраты на производство и хранение продукции, соответствующей этим месяцам, представлены на рисунке 5 в строчном варианте:

A	B	C	D	E	F	G	H	I	J
Yt									
6,00	12,00	0,00	3,00	4,00	11,00	0,00	15,00	0,00	5,00
Xt									
1,00	1,00	0,00	1,00	1,00	1,00	0,00	1,00	0,00	1,00
St									
0,00	7,00	0,00	0,00	0,00	2,00	0,00	7,00	0,00	0,00

Рисунок 5 – Результат задачи

3.2 Вторая модель

Вторая модель использует следующие переменные:

q_{it} — столько продукции производится в месяц i, чтобы выполнить заказ в месяц t, $t \geq i$, переменные x_t остались теми же, что и в предыдущей модели.

Используя введенные переменные, получаем следующую математическую модель:

$$min \sum_{t=1}^{T} \sum_{i=1}^{t} (p_i + h_i + h_{i+1} + \cdots + h_{t-1}) q_{it} + \sum_{t=1}^{T} c_t x_t, \quad (7)$$

$$\sum_{i=1}^{t} q_{it} = d_t,\ t = 1, \dots, T, \quad (8)$$

$$q_{it} \leq d_t x_i, i = 1, \dots T, t = i, \dots T, \quad (9)$$

$$q_{it} \geq 0, x_t \in \{0,1\}, t = 1, \dots, T. \quad (10)$$

Вторая модель обладает тем свойством, что если требование целочисленности переменных x_t заменить на условие непрерывности переменных $0 \leq x_t \leq 1$, то в оптимальном решении соответствующей задачи линейной релаксации переменные xt будут принимать значение 0 или 1, т. е. разрыв целочисленности для *LR* (7) – (10) равен нулю. Первая модель — это *MIP*, поскольку в ограничениях (4) имеются большие константы, то разрыв целочисленности не равен нулю.

4 Примеры задач

4.1 Задача о коммивояжёре

Одна из самых известных и важных задач транспортной логистики (и класса задач оптимизации в целом) – задача коммивояжера. Также встречается название: Задача о бродячем торговце. Суть задачи сводится к поиску оптимального, то есть кратчайшего пути, проходящего через некие пункты по одному разу. Например, задача коммивояжера может применяться для нахождения самого выгодного маршрута, позволяющего коммивояжеру объехать определенные города со своим товаром по одному разу и вернуться в исходную точку. Мерой выгодности маршрута будет минимальное время, проведенное в пути, минимальные расходы на дорогу или, в простейшем случае, минимальная длина пути.

Общий план решения задачи коммивояжера:

Для решения задачи коммивояжера методом ветвей и границ необходимо выполнить следующий алгоритм (последовательность действий):

1 Построение матрицы с исходными данными;

2 Нахождение минимума по строкам;

3 Редукция строк;

4 Нахождение минимума по столбцам;

5 Редукция столбцов.

6 Вычисление оценок нулевых клеток.

7 Редукция матрицы.

8 Если полный путь еще не найден, переходим к пункту 2, если найден к пункту 9.

9 Вычисление итоговой длины пути и построение маршрута.

Подробная методика решения задачи коммивояжера:

В целях лучшего понимания задачи будем оперировать не понятиями графа, его вершин и т.д., а понятиями простыми и максимально

приближенными к реальности: вершины графа будут называться города, ребра их соединяющие – дороги. Итак, методика решения задачи коммивояжера:

1 Построение матрицы с исходными данными:

Сначала необходимо длины дорог, соединяющих города, представить в виде следующей таблицы (Таблица 1):

Таблица 1 – Матрица исходных данных

Город	1	2	3	4
1	M	5	11	9
2	10	M	8	7
3	7	14	M	8
4	12	6	15	M

В нашем примере у нас 4 города и в таблице указано расстояние от каждого города к 3-м другим, в зависимости от направления движения (т.к. некоторые ж/д пути могут быть с односторонним движением и т.д.). Расстояние от города к этому же городу обозначено буквой М. Также используется знак бесконечности. Это сделано для того, чтобы данный отрезок путь был условно принят за бесконечно длинный. Тогда не будет смысла выбрать движение от 1-ого города к 1-му, от 2-ого ко 2-му, и т.п. в качестве отрезка маршрута.

2 Нахождение минимума по строкам:

Находим минимальное значение в каждой строке (di) и выписываем его в отдельный столбец. Результат заносим в таблицу 2.

Таблица 2 – Минимум по строкам

Город	1	2	3	4	di
1	M	5	11	9	**5**
2	10	M	8	7	7
3	7	14	M	8	7
4	12	6	15	M	**6**

3 Редукция строк:

Производим редукцию строк – из каждого элемента в строке вычитаем соответствующее значение найденного минимума (di). Результат заносим в таблицу 3.

Таблица 3 – Матрица, приведенная по строкам

Город	1	2	3	4	di
1	M	0	6	4	5
2	3	M	1	0	7
3	0	7	M	1	7
4	6	0	9	M	6

В итоге в каждой строке будет хотя бы одна нулевая клетка.

4 Нахождение минимума по столбцам:

Далее находим минимальные значения в каждом столбце (dj). Эти минимумы выписываем в отдельную строку. Результат заносим в таблицу 4.

Таблица 4 – Минимум по столбцам

Город	1	2	3	4	di
1	M	0	6	4	5
2	3	M	1	0	7
3	0	7	M	1	7
4	6	0	9	M	6
dj	0	0	1	0	

5 Редукция столбцов:

Вычитаем из каждого элемента матрицы соответствующее ему dj. Результат заносим в таблицу 5.

Таблица 5 – Матрица, приведенная по столбцам

Город	1	2	3	4	di
1	M	0	5	4	5
2	3	M	0	0	7
3	0	7	M	1	7
4	6	0	8	M	6
dj	0	0	1	0	

В итоге в каждом столбце будет хотя бы одна нулевая клетка.

6 Вычисление оценок нулевых клеток:

Для каждой нулевой клетки получившейся преобразованной матрицы находим оценку (Таблица 6 - 7). Ею будет сумма минимального элемента по строке и минимального элемента по столбцу, в которых размещена данная

нулевая клетка. Сама она при этом не учитывается. Найденные ранее di и dj не учитываются. Полученную оценку записываем рядом с нулем, в скобках.

Таблица 6 – Оценка нулевой клетки

Город	1	2	3	4
1	M	0 (4)	5	4
2	3	M	0	0
3	0	7	M	1
4	6	0	8	M

И так по всем нулевым клеткам:

Таблица 7 – Оценка всех нулевых клеток

Город	1	2	3	4
1	M	0 (4)	5	4
2	3	M	0 (5)	0 (1)
3	0 (4)	7	M	1
4	6	0 (6)	8	M

7 Редукция матрицы:

Выбираем нулевую клетку с наибольшей оценкой (Таблица 8). Заменяем ее на M (Таблица 9). Мы нашли один из отрезков пути. Выписываем его (от какого города к какому движемся, в нашем примере от 4-ого к 2-му).

Таблица 8 – Выбор клетки с наибольшей оценкой

Город	1	2	3	4
1	M	0 (4)	5	4
2	3	M	0 (5)	0 (1)
3	0 (4)	7	M	1
4	6	0 (6)	8	M

Ту строку и тот столбец, где образовалось две M полностью вычеркиваем. В клетку, соответствующую обратному пути ставим еще одну букву M (т.к. мы уже не будем возвращаться обратно).

Таблица 9 – Замена значения в клетке на M

Город	1	2	3	4
1	M	0 (4)	5	4
2	3	M	0 (5)	M
3	0 (4)	7	M	1
4	6	M	8	M

8 Если полный путь еще не найден, переходим к пункту 2, если найден к пункту 9:

Если мы еще не нашли все отрезки пути, то возвращаемся ко 2-му пункту и вновь ищем минимумы по строкам и столбцам, проводим их редукцию, считаем оценки нулевых клеток и т.д. Если все отрезки пути найдены (или найдены еще не все отрезки, но оставшаяся часть пути очевидна) – переходим к пункту 9.

9 Вычисление итоговой длины пути и построение маршрута:

Найдя все отрезки пути, остается только соединить их между собой и рассчитать общую длину пути (стоимость поездки по этому маршруту, затраченное время и т.д.). Длины дорог, соединяющих города, берем из самой первой таблицы с исходными данными. В нашем примере маршрут получился следующий: 4 → 2 → 3 → 1 → 4. Общая длина пути: L = 30.

Практическое применение задачи коммивояжера:

Применение задачи коммивояжера на практике довольно обширно. В частности, ее можно использовать для поиска кратчайшего маршрута при гастролях эстрадной группы по городам, нахождения последовательности технологических операций обеспечивающей наименьшее время выполнения всего производственного цикла [8].

4.2 Задача календарного планирования трех станков

Заданы *n* деталей d_i $(i=\overline{1,n})$, последовательно обрабатываемые на трех станках: R_1, R_2, R_3, причем технологические маршруты всех деталей d_i одинаковы. Обозначим a_i, b_i, c_i – длительность обработки деталей d_i на первом, втором и третьем станках соответственно. Необходимо определить такую очередность запуска деталей в обработку, при которой минимизируется суммарное время завершения всех работ *T*.

Можно показать, что в задаче трех станков очередность выполнения первых, вторых и третьих операций в оптимальном решении может быть

одинаковой (для четырех станков это свойство уже не выполняется). Поэтому достаточно определить очередность запуска только на одном станке (например, первом).

Обозначим $\sigma^k = (i_1, i_2, ..., i_k)$ – некоторую последовательность очередности запуска длиной k $(1 \le k \le n)$ и $A(\sigma^k)$, $B(\sigma^k)$, $C(\sigma^k)$ – время окончания обработки последовательности деталей σ^k на первом, втором и третьем станках соответственно.

Необходимо найти такую последовательность σ^* длины n, что $C(\sigma^*) = \min_{\sigma} C(\sigma)$.

Покажем, как можно рекуррентно вычислять величины $A(\sigma^k)$, $B(\sigma^k)$, $C(\sigma^k)$. Пусть $\sigma^{k+1} = \left(\sigma^k, i_{k+1}\right)$, т.е. последовательность деталей σ^{k+1} получена из последовательности σ^k добавлением еще одной детали i_{k+1}.

Тогда:

$$A(\sigma^{k+1}) = A(\sigma^k) + a_{i_{k+1}}, B(\sigma^{k+1}) = \max\left[A(\sigma^{k+1}), B(\sigma^k)\right] + b_{i_{k+1}},$$

$$C(\sigma^{k+1}) = \max\left[B(\sigma^{k+1}), (\sigma^k)\right] + c_{i_{k+1}}.$$

Алгоритм метода ветвей и границ для задачи календарного планирования трех станков:

Нулевой шаг:

Задается множество $G^{(0)}$, которое образовано всеми возможными последовательностями, т.е. $\sigma = \varnothing$. Вычисляется оценка для множества $G^{(0)}$

$$\Delta(\varnothing) = \max\left\{\delta_A(\varnothing), \delta_B(\varnothing), \delta_C(\varnothing)\right\},$$

где $\delta_A(\varnothing) = A(\varnothing) + \sum_j a_j + \min_j (b_j + c_j)$,

$$\delta_B(\varnothing) = B(\varnothing) + \sum_j b_j + \min_j c_j,$$

$\delta_C(\varnothing) = C(\varnothing) + \sum_j c_j$.

Первый шаг:

Производится разбиение множества $G^{(0)}$, т.е. $G^{(0)} = G_1^{(1)} \cup G_2^{(1)} \cup ... \cup G_n^{(1)}$. Подмножество $G_k^{(1)}$ определяется всеми последовательностями с началом i_k $(k = \overline{1,n})$.

Вычисление оценок:

Оценку для последовательности $\sigma_k^{(1)}$ определяют $\Delta(\sigma) = \max\left\{\delta_A(\sigma_k^1), \delta_B(\sigma_k^1), \delta_C(\sigma_k^1)\right\}$, $k = \overline{1,n}$.

Из всех подпоследовательностей выбирают наиболее перспективную последовательность $\sigma_k^{(1)}$ с наименьшей оценкой, т.е. $\Delta(\sigma_k^{(1)}) = \min_j \Delta(\sigma_j^{(1)})$.

Ветвление:

Перспективный вариант последовательности $\sigma_k^{(1)}$ развивают построением всех возможных подпоследовательностей длиной 2 с началом $\sigma_k^{(1)}$ вида $\sigma_{k+1}^{(2)} = \left(\sigma_k^{(1)}, j\right)$, где j не входит в $\sigma_k^{(1)}$.

k – й шаг:

Пусть проведено уже k шагов и построены варианты $\sigma_1^{(k)}$, $\sigma_2^{(k)}$,…,$\sigma_{\nu_k}^{(k)}$, т.е. подпоследовательности длиной k. Выбираем самый перспективный вариант $\sigma_s^{(k)}$, такой, что $\Delta(\sigma_s^{(k)}) = \min_j \Delta(\sigma_j^{(k)})$.

Множество $G_s^{(k)}$ разбиваем на (n – k) подмножеств, каждое из которых образуется добавлением к последовательности $\sigma_s^{(k)}$ некоторого элемента i_{k+1}, такого, что $i_{k+1} \notin \sigma_s^{(k)}$ [5].

В результате строим дерево вариантов следующего вида: вершина $G^{(0)}$

отвечает $\sigma = \varnothing$, вершины первого уровня $G_1^{(1)},...,G_{\text{n}}^{(1)}$ соответствуют последовательностям длиной 1, т. е. $\sigma_1^{(1)} = 1,...,\sigma_n^{(1)} = n$ и т.д. Каждая конечная вершина отвечает последовательности максимальной длины *n*.

Признак оптимальности:

Если $\sigma_v^{(n)}$ отвечает конечной вершине дерева, причем оценка $\Delta(\sigma_v^{(n)})$ наименьшая из оценок всех вершин, то $\sigma_v^{(n)}$ – искомый вариант. В противном случае переходим к продолжению варианта с наименьшей оценкой.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 Алексеева Е.В. Построение математических моделей целочисленного линейного программирования. Примеры и задачи: Учеб. пособие /Новосиб. гос. ун-т. Новосибирск, 2012.

2 Акимов О.Е. Дискретная математика. Логика, группы, графы, Москва, 2003, 376 с., ил., изд. дом «Лаборатория базовых знаний»

3 Новиков Ф.А. Дискретная математика для программистов, С.-Петербург, 2002 г. 304 с., ил., изд. дом «Питер»

4 Бондарев В.М. Основы программирования, 1998 г., 368 с. изд. дом «Феникс»

5 Кармазин В.Н., Шаповаленко В.В., Бреславцев Р.В. Дискретное и сетевое программирование: Практикум. Краснодар: Кубанский гос. Ун-т, 2006. – 242с.

6 Схрейвер А. Теория линейного и целочисленного программирования: в 2-х т. Т.2: Пер с англ. - М.: Мир, 1991. - 342с.

7 Оренбургский гос. Педагогический ун-т
URL: http://www.orenipk.ru/kp/distant_vk/docs/2_1_1/inf/inf_mat_mod.html (дата обращения 03.11.2015)

8 Сайт преподавателя экономики
URL: http://galyautdinov.ru/post/zadacha-kommivoyazhera (дата обращения 02.12.2015)

9 Файловый архив для студентов
URL: http://www.studfiles.ru/preview/1700241/ (дата обращения 01.11.2015)