ФЕДЕРАЛЬНОЕ АГЕНТСВО ПО ОБРАЗОВАНИЮ РОССИЙСКОЙ ФЕДЕРАЦИИ

УФИМСКИЙ ГОСУДАРСТВЕННЫЙ АВИАЦИОННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Общенаучный факультет

Кафедра ВВТиС

Отчет по лабораторным работам №5

**По дисциплине «Методы оптимизации»**

Группа ПМИ-440

Студент \_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_ Сайфутдинов Р.Ф.

(дата) (подпись) (Фамилия И.О.)

Проверил \_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_ Касаткин А. А.

(дата) (подпись) (Фамилия И.О.)

Уфа 2016

**Цель работы:** овладеть практическими навыками решения задач линейного целочисленного программирования методом Гомори.

**Постановка задачи**

Ассортимент выпускаемой продукции включает пастеризованное молоко, кефир и сметану, а также дополнительную продукцию согласно индивидуальному варианту задания.

* Затраты сырого молока составляют:
* На пастеризованное молоко – 1,01 кг/кг;
* На кефир – 1,01 кг/кг;
* На сметану – 9,45 кг/кг.
* Поставщики в состоянии поставить не более 140 ц молока в сутки.
* Фасовка молока и кефира осуществляется на автоматизированной линии производительностью 5 ц молока или 6 ц кефира в час. В течение суток линия может эксплуатироваться не более 21 часа.
* Фасовка сметаны осуществляется на другой автоматизированной линии производительностью 30 кг сметаны в час. В течение суток линия может эксплуатироваться не более 16 часов.
* Цена реализации пастеризованного молока – 2,4, кефира – 2,7, сметаны – 13,8 тыс. руб./ц.
* План должен обеспечивать максимальную выручку от реализации молочной продукции (контракт на поставку молока уже оплачен).
* Дополнительный вид продукции — йогурт. Цена — 2750 руб./ц. Затраты сырого молока — 0,95 ц/ц. Производительность фасовки на оборудовании для фасовки сметаны — 0,25 ц/ч. Максимальный выпуск — 15 ц/сут.
* Дополнительное ограничение: количество выпускаемой продукции считается в упаковках
* пастеризованное молоко — 1 кг/упаковка;
* кефир, йогурт — 500 г/упаковка;
* сметана, творог — 200 г/упаковка;
* творожные сырки — 100 г/упаковка.

Найти максимум функции:

где – количество производимых в сутки пакетов молока;

– количество производимых в сутки пакетов кефира;

– количество производимых в сутки упаковок сметаны;

– количество производимых в сутки пакетов йогурта.

Множество допустимых решений задается следующими условиями:

используя метод Гомори. Сравнить полученное значение с результатами решения задачи средствами Maple.

**Теоретическая часть**

*МЕТОД ГОМОРИ*

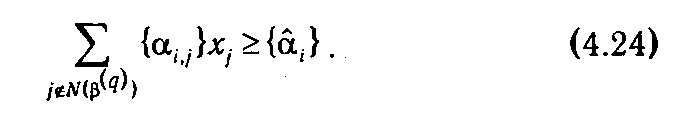
Данный метод, который также носит название метода отсекающих плоскостей, предназначен для решения ЦЗЛП в канонической форме.

*Описание алгоритма.* Приведем обобщенную схему алгоритма Гомори. Структурно он делится на так называемые большие итерации. Каждая большая итерация содержит этапы:

1. Решение «текущей» задачи методами линейного программирования (малые итерации). На первой итерации в качестве «текущей» задачи выступает нецелочисленный аналог исходной ЦЗЛП.

2. Определение первой нецелочисленной компоненты в оптимальном плане, полученном на этапе 1. Если все компоненты являются целочисленными, то алгоритм завершается.

3. Построение для найденной компоненты условия отсечения согласно правилу:



где добавление сформированного ограничения к системе ограничений текущей задачи, т. е. формирование новой текущей задачи. Переход на начало следующей большой итерации.

Можно доказать, что приведенный алгоритм конечен. Это означает, что на некотором шаге (итерации) будет найден целочисленный оптимальный план или обнаружен факт отсутствия допустимых целочисленных планов.

В качестве существенного замечания по поводу метода Гомори следует добавить, что при его практической реализации на ЭВМ следует считаться с ошибками округления, т, к. в условиях машинной арифметики практически ни один план не будет целочисленным. Кроме того, накапливающиеся погрешности могут внести возмущения в алгоритм и «увести» от оптимального целочисленного плана.

**Решение задачи в пакете Maple** **пакетом Optimization**

**> restart;**

**> with(Optimization, LPSolve): with(simplex, display):**

**> consts:={14000 >= 101\*x[1] + 101\*2\*x[2] + 945\*5\*x[3] + 95\*2\*x[4],**

**21\*6\*5 >= 6\*x[1] - 2\*5\*x[2],**

**16\*25\*30 >= 25\*5\*x[3] + 30\*2\*x[4],**

**1500 >= 2\*x[4] };**



**> display(consts);**



**> obj := 2.4\*x[1] + 2.7\*2\*x[2] + 13.8\*5\*x[3] + 2.75\*2\*x[4];**



**> LPSolve(obj, consts, assume = {integer, nonnegative}, maximize, depthlimit = 7);**



**Решение задачи в пакете Maple** **методом Гомори**

**> restart;**

**> with(simplex): with(numtheory):**

Функция поправки коэффициентов:

**> fixFracCoeff := proc(x)**

**local kf:**

**kf:=op(x)[1];**

**if (abs(kf-round(kf))<1e-7)then**

**round(kf)\*op(x)[2];**

**else**

**kf\*op(x)[2]**

**end;**

**end:**

Функция решения задачи линейного програмирования и нахождения отсекающей плоскости:

**> find\_max\_and\_plane := proc(obj, consts)**

**local bs, L, i, k, n, m, m\_el, pos, tmp\_plane, plane, res, eq, c:**

**res := maximize(obj, consts, NONNEGATIVE, 'NewConsts'):**

**bs := basis(NewConsts):**

**L := []: k := 1: n := nops(res): m := nops(bs):**

**for i from 1 to n do:**

**tmp\_plane := res[i]:**

**if lhs(tmp\_plane) = bs[k] then:**

**L := [op(L), frac(rhs(tmp\_plane))]:**

**k := k + 1:**

**if k > m then: break: end if:**

**end if:**

**end do:**

**m\_el,pos := ListTools:-FindMaximalElement(L, position):**

**if m\_el = 0 then:**

**printf("Задача решена!\n");**

**return res:**

**end if;**

**eq := rhs(NewConsts[pos]):**

**c := eval(eq,{seq(x[i]=0,i=1..n)});**

**eq := map(fixFracCoeff, eq-c);**

**eq := map(proc(x) abs(op(x)[1]-ceil(op(x)[1]))\*op(x)[2] end proc, eq);**

**eq := -(c - floor(c)) + eq;**

**return [res, eq]:**

**end proc:**

Функция решения целочисленной задачи линейного программирования на основе метода Гомори:

**> solve\_gomory := proc()**

**global n, obj, x, consts, max\_and\_plane:**

**local plane, i, N:**

**N := n:**

**for i from 1 to 10 do:**

**N := N+1: x := array(1..N):**

**printf("N = %d\n", N);**

**plane := x[N] = max\_and\_plane[2]:**

**consts := consts union {plane}:**

**max\_and\_plane := find\_max\_and\_plane(obj, consts);**

**if nops(max\_and\_plane) = N then**

**printf("Решение:");**

**#print(max\_and\_plane);**

**break**

**end if:**

**if N > 16 then**

**print(seq(evalf(max\_and\_plane[1][k]), k = 1..n));**

**else**

**print(evalf(max\_and\_plane[1]));**

**end if:**

**end do:**

**end proc:**

Ход решения:

**> n:=8; x := array(1..n):**



**> consts:={x[5] = 14000 - 101\*x[1] - 101\*2\*x[2] - 945\*5\*x[3] - 95\*2\*x[4],**

**x[6] = 21\*6\*5 - 6\*x[1] - 2\*5\*x[2],**

**x[7] = 16\*25\*30 - 25\*5\*x[3] - 30\*2\*x[4],**

**x[8] = 1500 - 2\*x[4] };**

**>**



**> display(consts);**

**obj := 240\*x[1] + 270\*2\*x[2] + 1380\*5\*x[3] + 275\*2\*x[4];**





**> max\_and\_plane := find\_max\_and\_plane(obj, consts);**



**> solve\_gomory();**

N = 9



N = 10



N = 11



N = 12



N = 13



N = 14



N = 15

Задача решена!

Решение:



**Вывод:** в ходе данной лабораторной работы овладели практическими навыками решения задач линейного целочисленного программирования методом Гомори.