ВЫЧИСЛИТЕЛЬНЫЕ СИСТЕМЫ
1975 год Выпуск 63

УДК 681.31:681.3.06

ОРГАНИЗАЦИЯ ФУНКЦИОНИРОВАНИЯ ОДНОРОДНЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ И СТОХАСТИЧЕСКОЕ ПРОГРАММИРОВАНИЕ

В.А.Павский, В.Г.Хорошевский

Рассматриваются в стохастической постановке следующие задачи: распределения сервисных программ между оперативной и внешней памятью, разбиения сосредоточенной однородной вычислительной системы (ОВС) на подсистемы в зависимости от спросов, организации функционирования сосредоточенных ОВС коллективного пользования (с произвольным числом терминалов).

Одной из основных тенденций развития вычислительной техники является построение мощных вычислительных систем коллективного пользования. Представляется перспективным использовать в качестве таких систем однородные вычислительные системы [1]. Работа преследует цель обратить внимание специалистов на возможность решения задач функционирования ОВС методами стохастического программирования [2,3]. Впервые такой подход был продемонстрирован в [4].

З а д а ч а 1*). Имеется ОВС, при эксплуатации которой используется набор сервисных программ: программы ввода/вывода информации, контрольные и диагностические тесты, стандартные и отладочные программы и т.д. Общий объем памяти, необходимый для хранения всех обслуживающих программ, составляет B ячеек. В оперативной памяти ОВС можно занять лишь b ячеек, $b < B$. Следовательно, часть сервисных программ объемом ($B - b$) ячеек

*) Постановка задачи 1 справедлива для любого вычислительного средства.

должна размещаться во внешней памяти (и на внешних носителях информации).

Спрос на различные сервисные программы при работе ОВС случаен, тем не менее считается, что оценка распределения вероятностей спроса на программы известна.

Учитывая объем оперативной памяти ОВС, параметры набора сервисных программ и спроса на каждую из них, требуется осуществить стохастически оптимальное распределение программ между оперативной и внешней памятью.

Итак, пусть $\{I_i\}$, $i=\overline{1,L}$, – набор сервисных программ, необходимых для работы ОВС; a_i – объем памяти, требуемый для I_i. Предположим, что t_i – потери*), которые возникают из-за отсутствия I_i в оперативной памяти ОВС. Допустим, что p_i есть вероятность того, что при работе ОВС для решения конкретной задачи потребуется программа I_i, $i=\overline{1,L}$ (значения оценки p_i устанавливаются статистически). Пусть, далее x_i $(i=\overline{1,L})$ – переменные, определяющие размещение программ, причем

$$x_i = \begin{cases} 1, & \text{если } I_i \text{ размещена в оперативной памяти ОВС;} \\ 0 & \text{– в противном случае.} \end{cases}$$

Требуется найти такое размещение программ в оперативной и внешней памяти, при котором средние потери минимальны.

Таким образом, имеем следующую задачу стохастического программирования: при ограничениях

$$\sum_{i=1}^{L} a_i x_i \le b, \qquad (1)$$

$$x_i = 0,1 \qquad (2)$$

и целочисленности b и a_i требуется найти

$$\min_{\{x_i\}} \left\{ Z = \sum_{i=1}^{L} f_i(x_i) = \sum_{i=1}^{L} (1-x_i)\, t_i\, p_i \right\}$$

и все оптимальные x_i^*, $i=\overline{1,L}$.

Для решения задачи используем динамическое программирование [2,3]. Заметим, что целевая функция Z – сепарабельна, остается преобразовать область допустимых значений для x_i, $i=\overline{1,L}$.

---

*) Потери (например, времени) – следствие того, что быстродействие внешних устройств и памяти существенно меньше быстродействия оперативной памяти.

Введем обозначения. Пусть $X=(x_1,\dots,x_i,\dots,x_L)$ – точка L-мерного евклидова пространства, а $X^*=(x_1^*,\dots,x_i^*,\dots,x_L^*)$ – наилучшее решение задачи. Пусть M_2 – множество таких X, которые удовлетворяют (1), (2), а M_1 – множество X, удовлетворяющих (1) и условию $x_i \geq 0$. Очевидно, что $M_1 \supset M_2$.

Рассмотрим функцию $Z_1(X)$, которая определяется условиями:

1) Z_1 совпадает с функцией Z на множестве M_2;

2) для каждого $X_1 \in M_1 \setminus M_2$ найдется $X_2 \in M_2$ такое, что $Z_1(X_1) > Z_1(X_2)$.

Из этих условий следует, что

$$\min_{X\in M_1} Z_1 = \min_{X \in M_2} Z.$$

Таким образом, если существует функция Z_1, удовлетворяющая вышеприведенным условиям, то возможность решения нашей задачи методом динамического программирования становится очевидной.

В самом деле, рассмотрим функцию

$$Z_1 = \sum_{i=1}^{L} t_i\, p_i\, |1-x_i|. \qquad (3)$$

Ясно, что условие 1 для нее выполняется. Докажем выполнение условия 2.

Пусть $X_1=(x_{11},\dots,x_{1i},\dots,x_{1L})$, $X_2=(x_{21},\dots,x_{2i},\dots,x_{2L})$. Если $X_1 \in M_1 \setminus M_2$, то для некоторых компонент вектора X_1 будет выполняться условие $x_{1i} > 1$. Определим X_2 следующим образом: если $x_{1i}=0$, то $x_{2i}=0$, если $x_{1i} \geq 1$, то $x_{2i}=1$. Тогда $X_2 \in M_2$.

Далее, $Z_1(X_1)$ отличается от $Z_1(X_2)$ только теми $f_i(x_i) = t_i p_i |1-x_i|$, у которых $x_{1i} > 1$, а $x_{2i}=1$. Для всех таких i выполняется с учетом (3) условие:

$$f_i(x_{2i}) < f_i(x_{1i}),$$

ибо в этом случае $f_i(x_{2i})=0$, а $f_i(x_{1i}) > 0$. Отсюда $Z_1(X_1) > Z_1(X_2)$, что и требовалось доказать.

Окончательно наша задача сводится к следующей: найти

$$\min_{\{x_i\}} \left\{ Z = \sum_{i=1}^{L} P_i t_i \,|\, 1 - x_i \,| \right\}$$

и все x_i^* при ограничениях:

$$\sum_{i=1}^{L} a_i x_i \le b, \quad x_i \ge 0, \quad i = \overline{1,L},$$

где b, a_i, x_i – целые числа.

Эту задачу, как показано в [3], можно решить методом динамического программирования. В приложении I приводится программа, являющаяся стержнем вычислительного процесса динамического программирования с сепарабельной целевой функцией и ограничениями вида (I) и $x_i \ge 0$, $i = \overline{1,L}$. В дальнейшем к стержню добавляются:

I. Массивы P и t, образуемые значениями P_i и t_i. Описываются и вводятся эти массивы во втором блоке стержня:

... array $t, P [1:n]$; read (a, t, P); ...

2. Процедура-функция $F(A,B)$, вычисляющая $f_i(x_i)$. Её описание имеет вид:

real procedure $F(A,B)$; integer AB; $F := t[A] P[A]\, abs(1-B)$, ...

Описание процедуры вводится в третьем блоке.

П р и м е р I. Имеется свободное поле оперативной памяти объемом I0 условных единиц. Имеются три обслуживающие программы. Первая программа требует 4 единицы памяти, время вызова её составляет 5 единиц времени, а вероятность вызова равна 0,I. Те же показатели для второй программы имеют значения 5; 6; 0,6; для третьей – 6; 7; 0,3. Таким образом, имеем: $L = 3$, $b = 10$, $a = (4;5;6)$, $t = (5;6;7)$, $P = (0{,}1;\ 0{,}6;\ 0{,}3)$. Требуется определить $X^* = (x_1^*, x_2^*, x_3^*)$ и

$$\min_{\{x_1, x_2, x_3\}} \left\{ Z = 0{,}5\,|1 - x_1| + 3{,}6\,|1 - x_2| + 2{,}1\,|1 - x_3| \right\}.$$

Получаем ответ $X^* = (1;1;0)$, $Z^* = \min Z = 2{,}1$. Это означает, что первая и вторая программы должны размещаться в оперативной памяти, а третья – во внешней. Минимум потерь составляет 2,I единиц времени.

З а д а ч а 2. Имеется ОВС, состоящая из n ЭМ. Программным способом [I] система может быть разбита по подсистемы различных рангов. (Подсистема имеет ранг j, если она состоит из ЭМ.) В общем случае разбиение системы может осуществляться таким образом, что некоторые из ЭМ не будут включены ни в одну из подсистем (например, это может быть обусловлено экономическими соображениями). Ясно, что может быть n различных рангов подсистем: из одной ЭМ, из двух ЭМ и т.д. Очевидно, что при разбиении ОВС на подсистемы должно выполняться условие

$$\sum_{j=1}^{n} j\, x_j \le n,$$

где x_j – число подсистем ранга j, $x_j = 0, 1, \ldots, [n/j]$; $[n/j]$ – целая часть числа n/j.

В различных подсистемах могут одновременно выполняться различные параллельные программы. Это позволяет организовать эффективное решение задач любой сложности (с любым количеством операций).

Будем считать, что ОВС предназначена для обслуживания потока задач различных рангов. (Задача имеет ранг j, если для её реализации требуется подсистема ранга $j = \overline{1,n}$.)

Проблема организации функционирования ОВС в случае управляемого потока задач (т.е. когда допустимо распределение задач по ЭМ в соответствии с определенной стратегией) решается [5] методами теории игр. Для неуправляемого потока задач проблема оперативного диспетчирования (разбиение ОВС на подсистемы и назначение имеющихся в данный момент времени задач) становится чрезвычайно сложной. Применение в данном режиме существующих алгоритмов распределения конечного множества задач по ЭМ системы [6] не будет оправданным. В самом деле, если использовать алгоритмы всякий раз, когда изменится обстановка на входе ОВС (как только поступят новые задачи), то положительный эффект распределения задач по машинам ОВС может быть скомпенсирован временем выполнения алгоритмов.

Покажем, что для неуправляемого потока задач проблема организации функционирования ОВС решается методами стохастического программирования.

Пусть α_j – цена, а c_j – стоимость эксплуатации [7] подсистемы ранга $j=\overline{1,n}$ в течение длительного промежутка времени (например, года). Допустим, что спрос a_j на подсистему ранга j в течение этого времени есть случайная величина, принимающая значения натурального ряда чисел, с известным распределением вероятностей $P_j(a_j)$. Тогда математическим ожиданием спроса на подсистему ранга $j=\overline{1,n}$ будет

$$b_j = \sum_{a_j=0}^{\infty} a_j P(a_j).$$

Если спрос на подсистему ранга $j=\overline{1,n}$ за время T превысит число выделенных подсистем ранга j, то убыток за каждый неудовлетворительный спрос составит $(\alpha_j - c_j)$. С другой стороны, если организовано подсистем ранга $j=\overline{1,n}$ больше, чем требуется, то убыток на каждую избыточную подсистему составит c_j.

Ожидаемые потери от недостатка подсистем ранга j составят величину

$$(\alpha_j - c_j)\sum_{a_j=x_j+1}^{\infty}(a_j - x_j)P_j(a_j), \tag{4}$$

а ожидаемые потери от избытка –

$$c_j \sum_{a_j=0} (x_j - a_j) P_j(a_j). \tag{5}$$

Математическое ожидание верхней границы прибыли от эксплуатации подсистемы ранга j равно

$$(\alpha_j - c_j)\sum_{a_j=0}^{\infty} a_j P_j(a_j) = (\alpha_j - c_j) b_j. \tag{6}$$

Математическое ожидание прибыли при эксплуатации подсистем ранга j выражается через величины (4)-(6) и равно

$$b_j(\alpha_j - c_j) - (\alpha_j - c_j)\sum_{a_j=x_j+1}^{\infty}(a_j - x_j)P_j(a_j) - c_j\sum_{a_j=0}^{x_j}(x_j - a_j)P_j(a_j).$$

Последнюю формулу легко преобразовать к виду

$$\theta(j) = \alpha_j b_j - c_j x_j - \alpha_j \sum_{a_j=x_j+1}^{\infty}(a_j - x_j)P_j(a_j).$$

Отсюда математическое ожидание прибыли при работе всех выделенных подсистем имеет вид

$$\sum_{j=1}^{n}\theta(j). \tag{7}$$

Требуется определить такие числа x_j^*, $j=\overline{1,n}$, которые соответствовали бы наибольшей ожидаемой прибыли за текущее время, то есть необходимо решить следующую задачу математического программирования:

найти

$$\min_{\{x_j\}}\left\{-\sum_{j=1}^{n}\left[\theta(j) - \alpha_j b_j\right]\right\} \tag{8}$$

и все оптимальные x_j^* при ограничениях: x_j – целые числа,

$$\sum_{j=1}^{n} j x_j \le n, \quad x_j \ge 0. \tag{9}$$

Это задача динамического программирования. В основе решения лежит программа, приведенная в приложении I. Дополнения следующие. Процедура – функция F, как и в задаче I, должна вычислять $f_j(x_j)$, имеющую вид

$$f_j(x_j) = c_j x_j + \alpha_j \sum_{a_j=x_j+1}^{\infty}(a_j - x_j)P_j(a_j).$$

Описание этой процедуры на языке АЛГОЛ-60:

```
real procedure F (A,B); integer A,B; begin integer u;real sum;
sum:=0; for u:=B+ 1 step 1 until m do; sum:=sum + (u-B)xP[A,u];
F:=c 1[A] + d 1[A] x sum end;
```

Переменной u соответствует a_j, что дана в постановке задачи; sum вычисляет значения

$$\sum_{a_j=x_j+1}^{\infty}(a_j-x_j)\,p_j(a_j).$$

Массив P, $c1$, $d1$ и число m – дополнительные данные, необходимые для вычисления $f_j(x_j)$.

В массивах $d2$ и $c2$ находятся d_j и c_j. Их описания и ввод находятся во втором блоке:

integer array $a, X_{opt}, d2, c2[1:n]$;... **read** $(a, d1, c1)$.

В массиве P находятся значения $P_j(a_j)$. Описание и ввод массива – также во втором блоке:

... **array** $P[1:n,\ 1:m]$; **read** $(a, P, d2, c2)$...

Таблица I

j	a_j: 0	I	2	x_j^*
I	0	0,I	0,9	2
2	0	0,5	0,5	2
3	0	0,8	0,2	I
4	0	0,6	0,4	I
5	0,2	0,8	0	I
6	0,5	0,5	0	0
7	0,9	0,I	0	0
8	0,4	0,6	0	0
9	I,0	0	0	0
I0	0,7	0,3	0	0
II	I,0	0	0	0
I2	I,0	0	0	0
I3	I,0	0	0	0
I4	I,0	0	0	0
I5	I,0	0	0	0
I6	0,8	0,2	0	0
I7	I,0	0	0	0
I8	I,0	0	0	0
I9	I,0	0	0	0
20	0,7	0,3	0	0

Массив P двумерный, в строке $j=\overline{1,n}$ находятся значения вероятностей спроса на подсистему ранга j. В каждой строке по m элементов: $P_j(1), P_j(2), \dots, P_j(m)$. Значение $P_j(0)$ не используется, так как из (4) следует, что $a_j \geq 1$ для всех x_j.

Назначение числа m следующее. Пусть a'_j – те a_j, для которых $P_j(a_j)$ является последним ненулевым значением вероятностей спроса на подсистему ранга j, тогда $m = \max\limits_{j\in\{1,\dots,n\}} a'_j$. Число m вводится и описывается в первом блоке:

... **integer** $n, b, j, s, x, i, d, c, m$

... **read** (n, b, m).

Пример 2. Имеется ОВС из 20 ЭМ. Это означает, что $j=\overline{1,20}$; $n=20$, $b=20$, $a_j=j$. Пусть d_1 равно двум денежным единицам, тогда $d_j=d_1\cdot j=2\cdot j$. Пусть c_1 равно одной денежной единице, т.е. $c_j=c_1\cdot j=j$, а распределение вероятностей спроса на подсистему ранга j представлено в табл.I.

Из табл.I видно, что $m=2$. Решением задачи будут:

$$X^* = (2,\ 2,\ I, I, I,\ 0,\ \dots,\ 0),\quad Z^* = 63{,}8.$$

То есть выделяются по две подсистемы рангов I и 2; по одной подсистеме рангов 3–5; а подсистем рангов 6–20 вообще не организуется. Относительные потери минимальны и составляют 63,8 денежных единиц.

В некоторых случаях знание конкретного закона распределения спроса на подсистему ранга j позволяет значительно упростить вычислительный процесс. Рассмотрим задачу 2, в которой распределение спроса a_j на подсистему ранга $j=\overline{1,n}$ подчинено закону Пуассона:

$$P(a_j) = \frac{b_j^{a_j}}{a_j!}\, e^{-b_j}, \qquad (I0)$$

где b_j – ожидаемая потребность в подсистеме ранга j.

Из (I0) следует, что

$$a_j P(a_j) = b_j P(a_j-1), \quad a_j \geq 1.$$

Тогда

$$\sum_{a_j=x_j+1}(a_j-x_j)P(a_j) = \begin{cases} b_j P(x_j-1) - x_j P(x_j), & \text{если } x_j \geq 1, \\ b_j, & \text{если } x_j = 0; \end{cases} \qquad (II)$$

$$P(x_j) = \sum_{a_j=x_j+1} P(a_j).$$

Соотношение (II) вместе с (8), (9) дает возможность без труда вычислять спрос на подсистему ранга j.

Поскольку этот частный случай важен сам по себе, в приложении 2 приводится программа на языке АЛГОЛ-60.

П р и м е р 3. Числовые значения параметров приведены в табл. 2.

Т а б л и ц а 2

	j										n
	I	2	3	4	5	6	7	8	9	I0	
c_j	2.4	3.5	3.7	3.9	4.0						5
a_j	I.3	2.0	3.0	3.4	3.6						
x_j^*	2	I	0	0	0						
c_j	3.I	2.7	3.5	3.4	4.0	8.5	5.2	6.4	6.8	6.9	I0
a_j	2.4	2.6	3.0	3.5	6.2	4.8	4.9	5.2	6.0	5.7	
x_j^*	2	I	I	0	0	0	0	0	0	0	

Так как оптимальное значение $z^* = \sum\limits_{j=1} a_j b_j + z$, то в примере 3 $z^* = 169{,}2$ при $n = 10$, $z^* = 79{,}9$ при $n = 5$.

З а д а ч а 3. Имеется сосредоточенная ОВС коллективного пользования. Пусть n – число ЭМ в ОВС, а ℓ – число терминалов (пультов). На каждый терминал поступает случайный поток задач различных рангов. Следовательно, спрос на подсистемы различных рангов для различных пультов не может быть заранее точно определен.

Пусть α_j – ранг подсистемы, которая требуется на пульте $j = \overline{1,\ell}$. Пусть также α_j – непрерывная*) случайная величина, имеющая плотность распределения вероятностей $P_j(\alpha_j)$; $\{\alpha_j\}$, $j = \overline{1,\ell}$ – независимые случайные величины. Обозначим через x_j ранг подсистемы, которая используется на пульте $j = \overline{1,\ell}$. Иначе говоря, α_j – число машин, которое требуется, x_j – число ЭМ, которое используется на пульте $j = \overline{1,\ell}$. Пусть d_j – цена эксплуатации, c_j – стоимость эксплуатации одной ЭМ для пульта $j = \overline{1,\ell}$.

Требуется так организовать функционирование ОВС, чтобы математическое ожидание функции потерь было минимально.

Оптимальные x_j, $j = \overline{1,\ell}$, являются решением следующей задачи:

*) Такое допущение правомерно, так как ОВС высокой производительности могут состоять из большого числа машин.

найти

$$\min_{\{x_j\}} \left\{ z = \sum_{j=1}^{\ell} \left[c_j x_j + d_j \int_{x_j}^{\infty} (\alpha_j - x_j) P_j(\alpha_j) d\alpha_j \right] \right\}$$

при ограничениях: x_j – целые числа,

$$\sum_{j=1} x_j \le n, \quad x_j \ge 0.$$

Решение задачи 3 осуществляется на основе программы приложения I при соответствующем изменении процедуры-функции $F(A,B)$.

Это далеко не полный перечень задач стохастического программирования. Тем не менее их достаточно, чтобы убедиться в целесообразности стохастического подхода. Достоинством рассматриваемого подхода является то, что он гарантирует стохастически оптимальное функционирование ОВС. Кроме того, трудоемкость вычислительных процедур, видимо, не является существенной. В самом деле, задачи организации стохастически оптимального функционирования ОВС, во-первых, могут быть решены при помощи исследуемой ОВС, во-вторых, они решаются только один раз на достаточно длительный период функционирования систем.

Уместно заметить, что стохастическое программирование целесообразно использовать и при организации функционирования распределенных ОВС [I] и вычислительных сетей [8].

Л и т е р а т у р а

I. ЕВРЕИНОВ Э.В., КОСАРЕВ Ю.Г. Однородные универсальные вычислительные системы высокой производительности. Новосибирск, "Наука", 1966.

2. ВАГНЕР. Основы исследования операций. Т. 3. М., "Мир", 1973.

3. ХЕДЛИ Дж. Нелинейное и динамическое программирование. М., "Мир", 1967.

4. ХОРОШЕВСКИЙ В.Г. Исследование функционирования однородных вычислительных систем. Диссертация на соискание ученой степени доктора технических наук. Л., 1973. (ЛЭТИ).

5. ХОРОШЕВСКИЙ В.Г., ТАЛНЫКИН Э.А. Теоретико-игровой подход к проблеме функционирования однородных вычислительных систем. -В кн.: Вычислительные системы. Вып. 5I. Новосибирск,1972, с. 20-27.

6. СЕДУХИНА Л.А., ХОРОШЕВСКИЙ В.Г.Стохастические алгоритмы функционирования однородных вычислительных систем. -"Автоматика и телемеханика", 1973, № 3, с. I2I-I28.

7. ХОРОШЕВСКИЙ В.Г., ХОРОШЕВСКАЯ Э.Г., ГОЛОСКОКОВА Т. М. Расчет технико-экономических показателей однородных универсальных вычислительных систем высокой производительности. - В кн.: Вычислительные системы. Вып. 39. Новосибирск, I970, с. 38-60.

8. Системы передачи данных и сети ЭВМ. М., "Мир", I974.

Поступила в ред.-изд.отд.
20 апреля I974 года

Программа динамического программирования

Вводятся следующие величины:

n - входной параметр (число ЭМ, число обслуживающих программ и т.д.);

b - число ограничений.

На печать выдаются:

Xopt - массив решений;

Z - оптимальное значение целевой функции.

Для того, чтобы запрограммировать определенную задачу, необходимо записать на входном языке вещественную процедуру-функцию с двумя формальными параметрами. Заголовок ее - F(A,B) . Эта процедура должна вычислять $f_j(x_j)$ по обращению к F(j,x) . Если для вычисления $f_j(x_j)$ необходимы дополнительные данные, то их нужно ввести и описать.

```
begin integer n,b,j,s,x,i,d,c; real r,Z; read (n,b);  begin
integer array a,Xopt[1:n], X1[1:n-1,0:b]; array L[1:n-1, 0:b]
read (a); c:=a[1]; for i:=2 step 1 until n do if a[i] < c  then
c:=a[i]; c:=entier(b/c);begin array f[0:c]; real procedure  Q ;
begin real r1; if j=1 then r1:=L[j-1,s-a[j]x x]; Q:=r1+f[x]end;
for j:=1 step 1 until n-1 do begin d:=entier(b/a[j]); for x:= 0
step 1 until d do f[x]:=F(j,x); for s:=0 step 1 until b do begin
d:=entier(s/a[j]); for x:=0 step 1 until d do begin if x=0 then
begin L[j,s]:=Q; X1[j,s]:=0; go to M1 end; r:=Q; M2: if r < L[j,
s] then begin L[j,s]:=r; X1[j,s]:=X; end; M1: end end end; j:=n;
s:=b; d:=entier(b/a[n]); for x:=0 step 1 until d do begin f[x]:=
F(n,x); if x=0 then begin Z:=Q; Xopt[n]:=0; go to M3 end; r:=Q;
M4: if r< Z then begin Z:=r; Xopt[n]:=x end; M3: end; d:=0; for
i:=1 step 1 until n-1 do begin d:=d+a[n-i+1]x Xopt[n-i+1]; Xopt
[n-i]:=X1[n-i,b-d] end; print ('Xopt', Xopt, 'Z=', Z) end end
end
```

Программа динамического программирования с пуассоновским законом распределения спроса на подсистемы различных рангов

Вводятся следующие величины:

n – число ЭМ в ОВС;

C5[1:n] – массив спросов на подсистему ранга j ;

D5[1:n] – массив ожидаемой потребности в подсистеме ранга j .

На печать выдаются:

X2 – число подсистем ранга j в порядке возрастания $j = \overline{1,n}$;

Z – оптимальное значение целевой функции (без ее постоянного члена).

```
begin integer n,i,j,b,c,s,C2;D1,D2,K2,K3; real G2,G3;  integer
array D,X2[1:n], X1[1:n-1, 0:n]; array F,C5,D5, F[1:n], LL[1:
n-1, 0:n]; procedure P(y2,y3); value y2; integer y2; real y3;
begin integer k,n; n:=1 for k:=1 step 1 until y2 do n:=n x k;
y3:=y2 ↑ y2 x exp(-y2)/n end; procedure FF(AA,y1); integer AA
real y1; begin real s1,s2,y1; integer j,n; array c1,S1[1:n];S1:
=s2:=0; for j:=1 step 1 until n do begin S1:=s1+c1[j] x AA;y1:=
AA+1; P(G,y1); s2:=s2+D1[j] x y1 end; y1:=s1+s2 end; real
procedure Q; begin integer n,b,j; real R1; array L[1:n-1, 0:n],
F[1:n]; if j:=1 then R1:=0 else begin R1:=L[j-1,s-j x b]; Q:=R1
+R1[j]  end end; print ('n=', n, c5, D5); begin for j:=1 step 1
until N-1 do begin D[j]:=entier (N/j); for x:=0 step 1  until
D[j] do begin FF(X,C2); F[j]:=G2 end; for s:=1 step 1 until  N
do begin D1:=entier(s/j);for x:=0 step 1 until D2 do begin if
x ≥ s then begin LL[j,s]:=Q; X1[j,s]:=x; goto M1 end else begin
R:=Q; goto M2 end; M2: if R < LL[j,s] then begin LL[j,s]:=R; X1
[j,s]:=X end; M1: end end end; k3:=N; s:=N; k3: I/S; for i:=0
step 1 until k3 do begin FF(k3,G3); k3[i]:=G3; if i=0  then
begin Z:=Q; X2[N]:=0; goto M3 end else begin R:=G; M4: if R < Z
then begin Z:=R; X2[N]:=i end end; M3: end; D1:=0; for i:=1 step
1 until N-1 do begin D1:=D1+(N-i+1) x X2[N-i+1]; X2[N-1]:=X1[N-
1, N-D1] end print (X2,Z) end end
```