ВОПРОСЫ ТЕОРИИ И ПОСТРОЕНИЯ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ
(Вычислительные системы)
1977 год
Выпуск 70

УДК 681.31:681.3.06

ОБ ОПТИМАЛЬНОМ РАСПРЕДЕЛЕНИИ МАШИН ПО ТЕРМИНАЛЯМ
РАСПРЕДЕЛЕННОЙ ОДНОРОДНОЙ ВЫЧИСЛИТЕЛЬНОЙ СИСТЕМЫ

В.И.Чиратков, Э.Г.Хорошеская

Предложен метод стохастически оптимального распределения машин по терминалам, учитывающий спрос на подсистемы различных рангов с терминалов для распределенных однородных вычислительных систем (ОВС) различной конфигурации: кольцевой, линейной, радиальной и произвольной.

Использование потенциальных возможностей распределенных ОВС в большой степени зависит от того, как организовано их функционирование, т.е. как осуществляется процесс решения задач на системе. В качестве показателей эффективности функционирования будем использовать целевую функцию, характеризующую потери при эксплуатации системы. Пусть распределенные ОВС коллективного пользования [1] состоят из n элементарных машин (ЭМ), размещенных на T вычислительных центрах (ВЦ), и обслуживают t терминалов, причем nj (j = 1, ..., T) - число ЭМ на i-м ВЦ, \( \sum_{i=1}^{T} n_i = n \), и i-й ВЦ принимает \( \ell_i \) терминалов, \( \sum_{i=1}^{T} \ell_i = \ell \).

Не терминалы поступают потоки задач различных рангов. Спрос на подсистемы различных рангов подчинен пуассоновскому распределению со средними значениями \( \mu_j \) min и \( \mu_j \) max, j = 1, ..., \ell, соответствующим при минимальном и максимальном требованиям рангов. Стоимость использования одной ЭМ с j-го терминала \( C_{ij} \), j = 1, ..., \ell. Матрица \( C_{ij} \) задает минимальные стоимости каналов между i-м и k-м ВЦ, i, k = 1, 2, ..., T.

Пусть \( \alpha_j \) (j = 1, ..., \ell) - ранг подсистемы, используемой j-м терминалом. Обозначим через \( S_{ij} \) (i = 1, ..., T, j = 1, ..., \ell) - число ЭМ j-го ВЦ, используемых j-м терминалом, и введем функцию \( \delta(S_{ij}) \) следующим образом:

\[
\delta(S_{ij}) = \begin{cases} 
0, & \text{если } S_{ij} = 0, \\
\ell_i, & \text{если } S_{ij} > 0
\end{cases}
\]

Тогда потери, связанные с использованием каналов, составляют

\[
\sum_{j=1}^{\ell} \sum_{i=1}^{T} \delta(S_{ij}) \cdot c_{ik_j} \cdot \kappa_j = 1, \ldots, T,
\]

где \( \kappa_j \) - номер ВЦ, содержащего j-й терминал.

Считаем, что избыточные ЭМ на j-м терминале простаивают, а при необоснованной j-го терминале минимально необходимым числом ЭМ все машины, входящие в этот терминал, простаивают, и за нерешенную задачу выплачивается штраф, пропорциональный ее сложности.

Задача, которую требуется решить, имеет следующий вид: найти

\[
\min Z = \sum_{j=1}^{\ell} \left( \sum_{i=1}^{T} \delta(S_{ij}) \cdot c_{ik_j} \cdot \alpha_j \left( \alpha_j, \mu_j \text{min} \right) C_{ij} \alpha_j + K \mu_j \text{max} \right) + \sum_{j=1}^{\ell} \alpha_j \left( \alpha_j - \mu_j \text{max} \right) + \frac{C_{ij} \left( \alpha_j \left( \alpha_j - \mu_j \text{max} \right) \right)}{\left( \mu_j \text{max} P_j \left( \alpha_j, \mu_j \text{min} \right) \right)} + \frac{C_{ij} \left( \alpha_j \left( \alpha_j - \mu_j \text{max} \right) \right)}{\left( \mu_j \text{max} P_j \left( \alpha_j, \mu_j \text{min} \right) \right)}
\]

при условиях:

\[
\begin{align*}
\alpha_j, S_{ij} & - \text{неотрицательные целые числа,} \\
\sum_{i=1}^{T} S_{ij} & = n_i, \quad i = 1, \ldots, T, \\
\sum_{i=1}^{T} S_{ij} & = \alpha_j, \quad j = 1, \ldots, \ell.
\end{align*}
\]

Здесь

\[
P_j(\alpha_j, \mu_j \text{max}) = \sum_{i=1}^{\ell_j} \frac{\mu_j \text{max} e^{\mu_j \text{max} \alpha_j}}{\gamma_j - \alpha_j}
\]

и

\[
\delta_j(\alpha_j, \mu_j \text{min}) = \begin{cases} 
1, & \text{если } \mu_j \text{min} P_j(\alpha_j, \mu_j \text{min}) > 0, \delta, \\
0, & \text{если } \mu_j \text{min} P_j(\alpha_j, \mu_j \text{min}) < 0, \delta,
\end{cases}
\]
$K_{\mu_{max}}$ штраф за нерешение задачи на $j$-м терминале, считаем, что штраф за нерешение задачи пропорционален ее сложности, $K$ — коэффициент штрафа.

Задачу (1)-(2) стохастического программирования с требованиями целесообразности и большим числом ограничений не представляется возможным точно решить известными методами математического программирования [2]. Кроме того, вероятностный характер задачи (1)-(2) не определяет алгоритм на поиск точного оптимума. Тем более, что при дискретизации по более ценным является быстрое и гарантированное получение хотя бы приближенного решения, иначе получение точного решения через большие промежутки времени без полной уверенности в успехе.

Задача (1)-(2) решается методом цепей Монте-Карло [3]. Она состоит в выборе случайным образом распределений, находящихся на некотором расстоянии от базового, пока не будет найдено лучшее; оно затем становится новым, и моделирование продолжается до тех пор, пока в течение некоторого времени не будет наблюдаться улучшение. Полученное таким образом распределение является хорошим, поэтому случайный поиск продолжается в его окрестностях.

Примем за базовое следующее распределение

$$N_i = \{q_{11}, \ldots, q_{1j}, \ldots, q_{1m}, \ldots, q_{ij}, \ldots, q_{ij}, \ldots, \}$$

где $x_j$ — число ЭМ, выделенных $j$-му терминалу,

$$x_j = \begin{cases} 1 & i > j, \\ 0 & i \leq j, \\ \end{cases}$$

$i$ — номер терминала, начиная с которого $\sum_{j=i}^{\infty} q_j = n$; $q_{ij}$ — число $i$-й терминал, $\alpha$ — целое число, выбиранное на основании $\alpha$.

Таким образом, последовательность $N_i$ состоит из $\ell$ числовых интервалов, разделенных $\ell-1$ нужем. Интервал по двум соседним числами соответствует определенному терминалу и несет в себе информацию о том, сколько ЭМ и с каких ВЦ выделено данному терминалу.

ПРИМЕЧАНИЕ. Следует заметить, что последовательность $N_i$ получена при условии, что спрос на терминалах превышает возможности системы, т.е. $\sum_{j=1}^{\infty} \mu_j \geq n$. Это наиболее вероятный случай. Если же $\sum_{j=1}^{\infty} \mu_j < n$, а $\sum_{j=1}^{\infty} \mu_j \geq n$, то $x_j$ примет вид:

$$x_j = \begin{cases} \mu_{max}, & i > j, \\ \mu_{min}, & i \leq j, \\ \end{cases}$$

наверное, если $\sum_{j=1}^{\infty} \mu_j < n$, то

$$x_j = \begin{cases} \mu_{max}, & i = 1, \ldots, \ell, \\ \mu_{min}, & i \geq \ell, \\ \end{cases}$$

далее, для полученного базового распределения $N_i$ вычисляется целевая функция $Z_i$. Затем получаем новое распределение на расстоянии $h$ от базового, $n \geq l \geq h \geq 1$. (Расстояние между новым распределением и базовым есть число породных номеров новой последовательности, которые не следуют за тем же номером, что и в базовой последовательности.) Всякий раз, когда найдено меньшее значение целевой функции, получаемое распределение сохраняется в качестве нового базового. Если сделано $g$ попыток без уменьшения целевой функции, то берется распределение с расстоянием $h/2$ от базового, и так далее, пока не будет найдено удовлетворительное распределение с расстоянием $h/2$ от базового.
В приложении приведена программа, написанная на языке АЛГОЛ. Алгоритм реализован на вычислительной машине БЭСМ-6. Ясно, что время счета существенно зависит от величин $n$, $l$, $q$, $h$. По сравнению с методом Монте-Карло, где время для выбора случайной переменной возрастает пропорционально $n$, для метода Пример окончательно $h$. В работе [3] показано, что любое решение может быть получено из базового за конечное число шагов, поэтому увеличение $q$, $h$ можно получить минимальное значение элементарной функции, с этой точки зрения вычисление.
тывного использования (рис.1), организованных в распределенные
1) систему с кольцевой структурой (рис.2),
2) одномерную систему (рис.3),
3) систему с радиальной структурой (рис.4),
4) систему с произвольной структурой (рис.5).

Однако число ЭМ в распределенной системе будет: \( n = 16 \) (\( n_1 = 1, n_2 = 2, n_3 = 4, n_4 = 2, n_5 = 3, n_6 = 3 \)). Она обслуживает 32 терминалов (\( \ell_1 = 32, \ell_2 = 3, \ell_3 = 5, \ell_4 = 6, \ell_5 = 1, \ell_6 = 4, \ell_7 = 7, \ell_8 = 6 \)). Соответствующие значения \( c_i, \mu \) приведены в табл.1; элементы \( c_{ik}, i, k = 1, \ldots, 7 \), матрицы \( C_{ik} \) определяются для каждой конкретной системы как минимальные стоимости каналов между \( i \)-м и \( k \)-м Ц. Матрицу \( C_{ik} \) легко получить из матрицы \( C \), в которой определены стоимости всех возможных каналов:

\[
C = \begin{bmatrix}
0 & 3 & 5 & 6 & 7 & 2 & 4 & 3 & 5 & 2 & 5 & 7 \\
3 & 5 & 0 & 3 & 5 & 2 & 5 & 4 & 3 & 5 & 5 \\
6 & 3 & 0 & 3 & 3 & 2 & 7 & 4 & 5 & 5 \\
7 & 2 & 5 & 3 & 3 & 0 & 3 & 2 & 8 & 3 & 5 \\
4 & 3 & 2 & 5 & 2 & 7 & 3 & 0 & 1 & 8 & 3 \\
5 & 2 & 4 & 3 & 4 & 2 & 8 & 1 & 8 & 0 & 1 & 5 \\
5 & 7 & 5 & 5 & 5 & 5 & 3 & 5 & 3 & 1 & 5 & 0 
\end{bmatrix}
\]

В табл.1 отражен результат назначения ЭМ распределенной ОСС по терминалам при \( h = 16, q = 500 \). Здесь же приведены соответствующие значения целевой функции \( Z^* \), а также \( \Delta Z = Z_{opt} - Z^* \), где \( Z_{opt} \) вычислено для исходного базового распределения. Время счета одного варианта системы на ЭМ БЭМ-6 составляет 15 мин. На рис.2-5 указаны только те терминалы, которым выделены ЭМ, в скобках указаны номера Ц, которыми пользуются соответствующие терминалы, например, на рис.5 [6] 3(н2),11 означает, что 10-й терминал выделен 3 ЭМ (две ЭМ с третьего Ц, одна с шестого); римскими цифрами указаны номера Ц.

По табл.2 можно проследить, как изменяются значения целевой функции от числа испытаний \( q \) и ростом \( h \) (и соответственно времени счета) для кольцевой распределенной ОСС. Очевидно, остатки \( q > 500 \) не имеют смысла, а при хорошем базовом распределении \( h \) может быть достаточно мало.

Достоинства метода цепей Монте-Карло станут очевидными, если учесть, что при полном переборе требуется вычисление целевой функции более чем для \( 2 \cdot 10^{15} \) различных комбинаций.
Табл. 2

<table>
<thead>
<tr>
<th>Расстояние от основной строки (м)</th>
<th>І</th>
<th>І,І</th>
<th>І,ІІ</th>
<th>І,ІІІ</th>
<th>І,ІV</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>11</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>12</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>13</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>14</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>15</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>16</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Таким образом, задача распределения машин по терминалам распределенной ОВС может быть поставлена в терминах стохастического программирования. Показано, что методы Монте-Карло являются при решении подобных задач. Данный подход достаточно просто позволяет организовать стохастически оптимальное функционирование распределенных ОВС.

Литература


Поступила в ред. отд. 26 января 1977 года
Программа для расчета стохастически оптимального распределения $\{x_1, \ldots, x_j, \ldots, x_k\}$ по терминалам и соответствующих потер $Z$.

Программа написана на языке АППЛ. Вводятся следующие исходные данные:

- $n$ - число ЭМ в распределенной ОВС,
- $l$ - число терминалов,
- $T$ - количество ВЦ,
- $N$ - коэффициент штрафов,
- $h$ - максимальное расстояние от базовой последовательности,
- $s_j$ - стоимость вкладывания одной ЭМ с $j$-го терминала,
- $u_{j\text{max}}$ - средний спрос на $j$-м терминале при максимально требуемом запасе,
- $u_{j\text{min}}$ - средний спрос на $j$-м терминале при минимально требуемом запасе.

$e_{i,k}$ - стоимость канала между $i$-м и $k$-м ВЦ $(i,k=1,\ldots,T)$,

$\mu_i$ - количество терминалов на $i$-м ВЦ $(i=1,\ldots,T)$,

$\eta_i$ - количество ЭМ на $i$-м ВЦ $(i=1,\ldots,T)$.

На печать выдаются значения:

- $J$ - номер терминала,
- $x_j$ - количество ЭМ, выделенных $j$-му терминалу,
- $s_j$ - количество ЭМ, выделенных $i$-ому ВЦ $j$-му терминалу.

```
begin integer n,l,h,s,i,j,p,q,x,T,S,D,m;
real d,p,q,j,k,m21[1:1],s1[1:T],r0,tau;
read (n,l,h,s,i,j,p,q,x,T,S,D,m);
begin real array cl,m1,m2[1:1],c,s[1:T],r2,m2[0:n],x,v,p,\lambda;
begin array m1,m2[1:n+1-],r2,m2[1:n+1-],\lambda,lambda[1:1],kappa[0:h],xmi[1:h],s[1:h,1:n+1-],PA[1:T];
begin procedure ДЕС (f,m1,m2,l,c,l,PA);
array f,m1,m2[l],c,l,real z;
begin z:=0; p:=d; i:=q:=0;
DE : S:=0; j:=1;
```

```
TY: if p > S then (j := j + 1; go to Q);
    i := 1; m := entier (mi2[p] + 0.5); x := q;
  
  WKB: if PA[N[i,j]] > 0 then
    begin Ml[x] := M[1,j]; PA[N[i,j]] := PA[N[i,j]] - 1;
        D := D + 1; if D = n then go to PAV
    end else (i := i + 1; if i = T then go to WKB);
    x := x + 1; if x = q + m - 1 then go to WKB;
    Ml[x] := 0; p := p + 1; q := x + 1; go to TY;
  
  N: if ro = 0; for i := 1 step 1 until l do ro := ro +
    enter(mi1[p] + 0.5); S := 0; p := q := j := 1;
  
  HEB: if p > S then (j := j + 1; go to WKB);
    l := 1; if ro ≤ 0 then m := entier (mi1[p] + 0.5);
    else m := entier (mi2[p] + 0.5); x := q;
  
  NW: if PA[N[i,j]] > 0 then
    (Ml[x] := M[1,j]; PA[N[i,j]] := PA[N[i,j]] - 1)
    else (i := i + 1; go to NW); x := x + 1;
    if x ≤ q + m - 1 then go to NW;
    Ml[x] := 0; p := p + 1; q := x + 1;
    if p < l then go to HEB else
    begin
      i := 1;
      PIV: if PA[1] > 0 then for x := q step 1 until q + PA[1] - 1 do
        Ml[x] := 1 else (i := i + 1; if i = T then go to PIV)
    end;
    PAV: if i = 1 step 1 until l do
    begin tau := u0 + u1; u0 := u1; if tau > 4 then tau := tau - 4;
        u1 := tau; v[1] := tau/4
    end;
    for i := 1 step 1 until l do X[i] := (n+1-d) x v[i]; p := 1;
    WKB: x := 1; if i = 1, ..., h do
    begin if X[p] = X[i] then (if p > 1 then x := x + 1)
    else (if X[p] > X[i] then x := x + 1)
    end;
    r(x) := X[p]; if p < h then
        (p := p + 1; go to WKB);
    for i := 1 step 1 until l do kappa [x] := entier (r(x));
    kappa[0] := 0; kappa[h] := n + 1; for l := 1 step 1 until l do
    begin tau := u0 + u1; u0 := u1; if tau > 4 then
        tau := tau - 4; u1 := tau; v[1] := tau/4
    end;
    p := 1;
  
  HKB: x := 1; for i := 1 step 1 until h do
  begin if v[p] = v[i] then (if p > 1 then x := x + 1)
  else (if v[p] > v[i] then x := x + 1)
  end;
  r(x) := x; kappa(x) := kappa[p] - kappa[p-1];
  if p < h then (p := p + 1; go to HKB);
  for i := 1 step 1 until h do
  begin if kappa[i] > kappa[i-1] then
    begin for j := kappa[i-1] step 1 until kappa[i]
    do s[i,j,kappa[i-1]] := Ml[j]
    end;
    
  end;
  
  j := 1; x := 1;
  R: if kappa[x] > 0 then
  for i := 1 step 1 until kappa[x] do
    (M[i,j] := q[i,j]; j := j + 1);
  if x < h then (x := x + 1; go to R);
  PIV: for i := 1 step 1 until l do
    begin Ml[i] := M2[i]; D := 0;
        for i := 1 step 1 until l do L[i] := lambda[i]
    end;
    D := D + 1; if D < g then go to HKB else (h := h/2; D := 1;
  if h > 1 then go to HKB); q := 1; for j := 1 step 1 until l do
  begin for i := 1 step 1 until T do
    if M[i] = 0 then print (j, i, PA[i]) else
    begin for p := q step 1 until q + L[j] - 1 do
      begin for x := 1 step 1 until T do
        if M[i] = 0 then PA[x] := PA[x] + x
      end;
      print (j, L[j], PA)
    end;
    q := q + L[j] - 1
  end end end end.