## 3.3. Алгоритм оптимальной компоновки программного ресурса для однородной системы конкурирующих процессов.

При разработке системного и прикладного программного обеспечения многопроцессорных систем возникают задачи эффективного управления множеством процессов, имеющих доступ к общим ресурсам, в том числе к программам [1]. Одним из методов решения таких задач является так называемый метод структурирования программных ресурсов на блоки с целью их параллельного использования множеством конкурирующих процессов [2].

Организация вычислений по этому методу позволяет существенно сократить суммарное время последовательного использования программного ресурса конкурирующими процессами путем конвейеризации выполнения блоков при линейном структурировании программного ресурса. В [3, 4] показано, что наибольший выигрыш достигается при равномерном структурировании на число блоков, удовлетворяющих определенным соотношениям. В тоже время при имеющемся линейном структурировании программного ресурса общее время его использования можно уменьшить за счет компоновки отдельных блоков программного ресурса в один программный блок.

В данной главе предлагается алгоритм построения оптимальной компоновки линейно структурированного программного ресурса, требующий не более $O(s^3)$ элементарных операций, где s – число блоков исходного структурирования.

## Основные понятия и постановка задачи

Пусть, как и в [3, 4], $P_r = (Q_1, Q_2, ..., Q_s)$ – линейно структурированный на s блоков (s≥2) программный ресурс, $\gamma_s = (t(Q_1), t(Q_2), ..., t(Q_s)) = (t_1, t_2, ..., t_s)$ – последовательность времен выполнения блоков линейно структурированного программного ресурса $P_r$, τ – время , характеризующее дополнительные системные расходы по организации структурирования и параллельного использования блоков программного ресурса $P_r$, n – число однородных относительно программного ресурса $P$, конкурирующих процессов (n ≥2), p – число однородных процессов (p≥2).

Тогда общее время выполнения n конкурирующих процессов, использующий программный ресурс $P_r$ в вычислительной среде с p однородными процессорами при асинхронном режиме взаимодействие процессов и в режиме непрерывного выполнения $P_r$, составляет величину [4].

$$T(p,n,\gamma_s)=\begin{cases} T^{'}+(\mathrm{n}-1)t_{*}^{'}, npu\ \ s=kp, \\ или\ \ p=n,\ \ T^{'}\le pt_{*}^{'}, \\ (k+1)T^{'}+(r-1)t_{*}^{'}, \\ npu\ \ p=n,\ \ T^{'}>pt_{*}^{'}, \end{cases} \qquad (1)$$

Здесь $T^{'}=\sum_{j=1}^{s}t_j+s\tau,\ \ t_{*}^{'}=\max_{1\le j\le s}\{t_j\}+\tau,\ \ n=kp+r,\ \ 1\le r\le p.$

С понятием линейного структурирования программного ресурсы свяжем понятие линейной упаковки.

Определение. Пусть $M=\{m_1,\ m_2,\ldots,m_s\}$ - конечная упорядоченное множество предметов разбиение множества M на l непересекающихся подмножеств $M_1$, $M_2$…, $M_l$ , такое что каждая $M_i$ есть объединение последовательных элементов множество M, будем называть линейной упаковкой множество M ранга l.и. обозначать $q_l$.

Элементы множества $M$ будем рассматривать как последовательность блоков линейно структурированного программного ресурса $T^{'}=p(\mathrm{t}^{*}(q_l)+\tau)$ в этом случае Линейная упаковка множество $M$ получается объединением последовательных блоков исходного структурирования в один программный блок линейную упаковку программного ресурса $P_r$ будем называть компоновкой.

Обозначим $\mathcal{P}$ , $\mathcal{P}_l$ соответственно множество всевозможных компоновок $P_r$ и компоновок $P_l$ ранга $l(\mathrm{l}=\overline{1,\mathrm{s}})$. Отметим, что что компоновкой ранга s является исходное структурирование $\mathrm{q}_s=(\mathrm{Q}_1,\mathrm{Q}_2,...,\mathrm{Q}_\mathrm{s})$, ранга 1 - компоновка блоков $P_r$ в один программный

блок $q_s = (Q_1 \cup Q_2 \cup ... \cup Q_s)$ Нетрудно подсчитать, что $|\mathcal{P}|=2^{s-1}$, $|\mathcal{P}| = C_{s-1}^{l-1} = \frac{(s-1)!}{(l-1)!(s-l)!}$.

Пусть $q_s = (Q_1', Q_2', ..., Q_s')$ - компоновка программного ресурса $P_r$. Обозначим $t(Q_j') = \sum_{Q \in Q_j'} t(Q)$ – время выполнения j-го элемента компоновки $q_l$ $(j = \overline{1,s})$, $\gamma(q_1) = (t(Q_1'), t(Q_2'), ..., t(Q_l'))$ последовательность времен выполнение блоков $Q_j'$, $i^*(q_i) = \max_{1 \le j \le l} \{t(Q_i')$ время выполнения максимального блока компоновки $q_j'$, $t_l^0 = \min\{t(q_l) \mid q_l \in P_l\}$

Задача оптимальной компоновки линейно структурированного программного ресурса равняется $P_r = (Q_1, Q_2, ..., Q_s)$ состоит в том чтобы при заданных $\tau, p \ge 2, n \ge 2, \gamma_s, s \ge 2$ найти и компоновку $q_l$ исходного структурирования при которой достигается минимум функционала (1). Такую компоновку будем называть оптимальной.

# Свойства оптимальных компоновок и вспомогательные результаты

Для решения поставленной задачи потребуются следующие результаты.

**Теорема 1.** Если $q_l$ оптимальная компоновка программного ресурса $P_r$, то компоновка $q_l'$ такая что $i^*(q_l') = t_l^0$ также является оптимальной.

**Доказательство.** Из определения $t_l^0$ следует что

$$i^*(q_l) = t_l^0 \quad (2)$$

Покажем, что при выполнении условий теоремы компоновка $q_l^{'}$ оптимальная т. е. при заданных $\tau, p, n$

$$T(\mathrm{p,n},\gamma(\mathrm{q}_l)) = T(\mathrm{p,n},\gamma(\mathrm{q}_l^{'})) \quad (3)$$

При $n = p$ для оптимальной компоновки $q_l$ $i^*(q_l) = t_l^0$ (2) и теорема выполняется. Пусть $p < n, n = kp + r, 1 \le r \le p$. Рассмотрим возможные случаи.

i) $T^{'} \le p(\mathrm{t}_l^0 + \tau), T^{'} \le p(\mathrm{t}^*(q_l) + \tau)$, тогда из (1) и условия оптимальности $q_l$ следует

$$T(\mathrm{p,n},\gamma(\mathrm{q}_l^{'})) - T(\mathrm{p,n},\gamma(\mathrm{q}_l)) = T^{'} + (\mathrm{n}-1)(\mathrm{t}_l^0 + \tau) - T^{'} - (\mathrm{n}-1)(\mathrm{t}^*(q_l) + \tau) =$$
$$(\mathrm{n}-1)(\mathrm{t}_l^0 - \mathrm{t}^*(q_l)) \ge 0$$

Отсюда в силу (2) следует, что $i^*(q_l) = t_l^0$ (3) выполняется.

ii) $T^{'} > p(\mathrm{t}_l^0 + \tau), T^{'} > p(\mathrm{t}^*(q_l) + \tau)$, тогда

$$T(\mathrm{p,n},\gamma(\mathrm{q}_l^{'})) - T(\mathrm{p,n},\gamma(\mathrm{q}_l)) = T^{'}(k-1) + (r-1)(\mathrm{t}_l^0 + \tau) - T^{'}(k-1) - (r-1)(\mathrm{t}^*(q_l) + \tau) =$$
$$(r-1)(\mathrm{t}_l^0 - \mathrm{t}^*(q_l)) \ge 0$$

Отсюда в силу (2) и $r \ge 1$ следует (3).

iii) $T^{'} > p(\mathrm{t}_l^0 + \tau), T^{'} \le p(\mathrm{t}^*(q_l) + \tau)$, тогда

$$T(\mathrm{p,n},\gamma(\mathrm{q}_l^{'})) - T(\mathrm{p,n},\gamma(\mathrm{q}_l)) = T^{'}(k+1) + (r-1)(\mathrm{t}_l^0 + \tau) - T^{'} - (\mathrm{n}-1)(\mathrm{t}^*(q_l) + \tau) =$$
$$\mathrm{kT}^{'} + (r-1)(\mathrm{t}_l^0 + \tau) - (\mathrm{kp} + \mathrm{r} - 1)(\mathrm{t}^*(q_l) + \tau) = \mathrm{k}(T^{'} - p(\mathrm{t}^*(q_l) + \tau) + (r-1)(\mathrm{t}_l^0 + \mathrm{t}^*(q_l)) \ge 0$$

что возможно лишь при $r = 1$ и $T^{'} = p(\mathrm{t}^*(q_l) + \tau)$. Отсюда следует справедливость (3).

Теорема доказана.

**Теорема 2**. Если для компоновок $q_l$, $q_{l-1}$ $(l < 2)$ $i^*(q_l) = i^*(q_{l-1})$, то

$$T(\mathrm{p}, \mathrm{n}, \gamma(\mathrm{q}_l) > T(\mathrm{p}, \mathrm{n}, \gamma(\mathrm{q}'_{l-1}))$$

Из теоремы 1 следует, что если для каждого ранга $l = 2,..., \mathrm{s},$ можно «эффективно» строить компоновку $q_l$ программного ресурса $P_r$ с наименьшим максимальным элементом среди компоновок этого ранга $(i^*(q_l) = t_l^0)$, то «эффективно» будет решена исходная задача, поскольку в этом случае оптимальную компоновку необходимо будет выбрать в худшем случае среди (s — 1) компоновки. Очевидно также, что наименьший максимальный элемент среди компоновок ранга $l$ с убыванием $l$ не убывает, т. е.

$$t_{l_1}^0 \geq t_{l_2}^0 \text{ при } 1 < l_1 < l_2 \leq s. \qquad (4)$$

С практической точки зрения является естественным предположение

$$\tau \leq t_l \ , \qquad i = \overline{1, \mathrm{s}}, \qquad (5)$$

что позволяет при решении задачи оптимальной компоновки исключить из рассмотрения компоновку в один программный блок.

Наряду с исходной задачей рассмотрим следующую оптимизационную задачу «линейной упаковки в контейнеры».

Для заданных предметов конечного упорядоченного множества $M = \{m_1, m_2, \ldots, m_s\}$ и соответствующей последовательности их размеров $v(m_1), v(m_2), \ldots, v(m_s)$ $(v(m_i) > 0)$, числа B > 0 — вместимости контейнера $(\mathrm{B} \geq \max\limits_{1 \leq i \leq s} \{v(\mathrm{m}_i)\})$, требуется найти такую линейную упаковку

$q_l = (M_1, M_2, \ldots\ldots, M_l)$ множества M, чтобы размер каждого элемента упаковки $v(M_i)$ не превосходил B и $l$ было наименьшим.

В общем случае, т. е. когда отсутствует условие линейности упаковки, эта задача является NP-трудной в сильном смысле, поскольку при $v(m_i) \in (0,1) \quad (i = \overline{1,\mathrm{s}})$ B = 1 дает классическую оптимизационную задачу упаковки в контейнеры [5]. Условие линейности упаковки, связанное с задачей оптимальной компоновки линейно структурированных программных ресурсов, существенно упрощает ее решение.

Задача линейной упаковки в контейнеры эффективно решается с помощью следующего *LF*-алгоритма (last—fit).

1. Первый предмет $m_1$ загружается в первый контейнер, а остальные предметы — в порядке возрастания их номеров.

2. Предмет $m_i \quad (i = \overline{2,\mathrm{s}})$ загружается в последний контейнер из числа частично упакованных, если сумма помещенных в него предметов не превосходит $\mathrm{B} - v(\mathrm{m}_i)$, в противном случае он загружается в следующий пустой контейнер.

Оптимальность линейной упаковки, которую строит *LF* - алгорнтм, легко доказывается методом от противного. *LF* -алгоритм требует не более 3s элементарных операций и является составной частью алгоритма решения исходной задачи оптимальной компоновки.

АЛГОРИТМ ПОСТРОЕНИЯ ОПТИМАЛЬНОЙ КОМПОНОВКИ

Пусть $\gamma_s = (t_1, t_2, \ldots, t_s)$ — последовательность времен выполнения блоков заданного линейно структурированного программного ресурса $\mathrm{P}_r = (\mathrm{Q}_1, \mathrm{Q}_2, \ldots, \mathrm{Q}_s) \quad (s \geq 3)$ n—число конкурирующих процессов, p —

число однородных процессоров, $\tau$ — время, характеризующее дополнительные системные расходы по организации структурирования на каждый блок ($\tau \le t_i,\ i = \overline{1,s}$).

Алгоритм построения оптимальной компоновки программного ресурса $P_r$ состоит из следующих этапов.

1. Строим массив из $\frac{s(s+1)}{2} - 1$ чисел $t_{ij} (i = \overline{2,s}, j = \overline{1,i})$ по правилу:

$$
\begin{aligned}
& t_{sj} = t_j,\ \ j = \overline{1,s}, \\
& \text{.....} \\
& t_{s-1\,j} = t_{sj} + t_{j+1},\ \ j = \overline{1,s-1}, \\
& \text{.....} \\
& t_{s-1\,j} = t_{sj} + t_{j+1},\ \ j = \overline{1,s-1}, \\
& t_{s-k,j} = t_{s-h+1,j} + t_{j+k},\ \ j = \overline{1,s-k}, \\
& \text{.....} \\
& t_{ij} = t_{sj} + t_{i=s-2},\ \ j = 1,2..
\end{aligned}
\qquad (6)
$$

Здесь числа $t_{ij} = t_{sj} + t_{i+1} + ... + t_{j+s-1}$ представляют собой длительности блоков всевозможных линейных компоновок $P_r$.

2. Упорядочиваем числа $t_{ij}$ по возрастанию с одновременным удалением избыточных одинаковых элементов и элементов $t_{ij} \le \max\limits_{1 \le i \le s} \{t_j\}$ В результате получим возрастающую последовательность чисел $v_1 < v_2 < v_3 < ... < v_k$, для которой $v_1 \le \max\limits_{1 \le i \le s} \{t_j\}$ , $s - 1 \le k < \frac{s(s+1)}{2} - 1.$

3. Полагаем $T_0 = T(p, n, \gamma_s),\ \gamma_0 = \gamma_s,\ l_0 = s,\ i = 1.$

4. Принимая вместимость B равной $v_i$, к исходному структурированию применяем *LF* -алгоритм линейной упаковки. Пусть $l_i$ — ранг полученной компоновки $P_r$

5. Если $l_i = l_{i-1}$, то полученную компоновку $q_i$ не принимаем в рассмотрение, вычисляем $i = i + 1$ и переходим к п. 4

6. Вычисляем значение $T_0 = T(p, n, \gamma(\mathrm{q}_{ij}))$. Если $T_i < T_0$, то полагаем $T_0 = T_i$, $\gamma_0 = \gamma(q_{ij})$, иначе $T_0$, $\gamma_0$ оставляем без изменений.

7. Если $l_i > 2$, то вычисляем i = i + 1 н переходим к п. 4, иначе ($l_i = 2$) алгоритм заканчивает работу.

После окончания работы алгоритма $T_0$ будет давать минимальное значение функционала (1), $\gamma_0$ — оптимальную компоновку $P_r$. Правильность его работы следует из теорем 1, 2 и соотношений (4), (5).

Приведенный алгоритм требует не более $O(s^3)$ элементарных операций, поскольку на первом этапе для построения $\mathrm{t}_{ij}$ требуется $O(s^3)$ элементарных операций, на втором, используя быстрые алгоритмы сортировки, — $O(s^3 \log_2 s)$, на этапах 4 в цикле по $v_i$ — не более $O(s^3)$.

Пример. Пусть $\gamma_{10} = (10, 20, 5, 20, 5, 10, 10, 5, 10, 5)$ — последовательность времен выполнения блоков линейно структурированного программного ресурса $\mathrm{P}_r = (\mathrm{Q}_1, \mathrm{Q}_2, ..., \mathrm{Q}_{10})$, n — 5 -- число конкурирующих процессов, p = 3 — число однородных

процессоров, i = 4 — дополнительнь1е системные накладные расходы на каждый блок, связанные со структурированием $P_r$. Поскольку в этом случае:

$$T' = \sum_{j=1}^{s} t_j + s\tau = 100 + 10 \cdot 4 > p(t_* + \tau) = 3(20 + 4),$$

то общее время использования программного ресурса $P_r$ заданным числом конкурирующих процессов (1) составит величину $T_{10} = T(3,5,\gamma_{10}) = 2(100 + 10 \cdot 4) + 1(20 + 4) = 304$ T10 = T (3, 5, "10) = 2 (100 + 10.4) + 1 (20 + 4) _ = 304 единиц времени. Найдем компоновку программного ресурса $P_r$, при которой достигается минимум функционала (1) с помощью приведенного выше алгоритма.

1. Строим массив из $\frac{10-11}{2} - 1 = 54$ чисел $t_{ij}$ $(i = \overline{2,10},\ j = \overline{1,i})$, согласно правилу (6). На рис. 1 дана схема формирования этих чисел. Они представляют собой сумму чисел, стоящих у основания соответствующего треугольника, образованного выходящими из $t_{ij}$ стрелками.

2. Упорядочивая $t_{ij}$ по возрастанию с одновременным удаением избыточных одинаковых элементов и элементов $t_{ij} < 20$, получим следующую возрастающую последовательность чисел

20, 25, 30, 35, 40. 45, 50, 55, 60, 65, 70, 75, 80, 85, 90, 95. (7)

Принимая последовательно вместимость B равной значениям элементов последовательности (7), к исходному структурированию применяем LF-алгоритм до тех пор, пока он не даст компоновку ранга 2. Вычисления, проводимые на каждом шаге этого этапа, приведены в

табл. 1, где прочерк означает, что нет необходимости вычислять соответствующее значение $T_0 = T(p, n, \gamma(\mathrm{q}_i))$. Как видно из табл. 1, оптимальной компоновкой будет

$\mathrm{q}_4 = ((Q_1 \cup Q_2), (Q_3 \cup Q_4 \cup Q_5), (Q_6 \cup Q_7 \cup Q_8), (Q_9 \cup Q_{10}))$, для которой у $\gamma(\mathrm{q}_4) = (30, 30, 25, 15)$, $T(3,5\gamma(q_5)) = 256$.

Таким образом, общее время использования исходного линейно структурированного программного ресурса P улучшено за счет оптимальной компоновки на 304 — 266 = 38 единиц времени или на $\frac{38}{308} \cdot 100 \approx 12\,\%$.

| B | $\gamma(\mathrm{q}_i)$ | | $T(3,5\gamma(q_5))$ |
|---|---|---|---|
| 20 | (10, 20, 5, 20, 15, 15, 15) | 7 | 280 |
| 25 | (10, 25, 25, 10, 25, 5) | 6 | 277 |
| 30 | (30, 30, 25, 15) | 4 | 266 |
| 35 | (35, 35, 30) | 3 | 268 |
| 40 | (35, 35, 30) | 3 | - |
| 45 | (35, 35, 30) | 3 | - |
| 50 | (35, 45, 20) | 3 | - |
| 55 | (55, 45) | 2 | 344 |

**Теорема.** Для того, чтобы эффективная одинаково распределенная система конкурирующих процессов в случае ограниченного параллелизма в асинхронном и втором синхронном режимах была оптимальной при заданных $p \geq 2,\ T_\varepsilon^n,\ \varepsilon > 0$, необходимо и достаточно, чтобы она была стационарной и число процессов $n_0$ в системе равнялось одному из чисел:

$$1)\ \left[\left[\sqrt{\frac{(p-1)T_\varepsilon^n}{k\varepsilon}}\right],\ \left[\sqrt{\frac{(p-1)T_\varepsilon^n}{k\varepsilon}}\right]+1\right] \cap [2,n],\ \text{при}\ s = kp,\ k > 1,$$

$$2)\ \left[\left[\sqrt{\frac{(r-1)T_\varepsilon^n}{(k+1)\varepsilon}}\right],\ \left[\sqrt{\frac{(r-1)T_\varepsilon^n}{(k+1)\varepsilon}}\right]+1\right] \cap [2,n],\ \text{при}$$

$s = kp + r,\ k \geq 1,\ 1 \leq r < p,$

в котором функция $\overline{\Delta}_\varepsilon(x) = (s-1)T^n\left(1 - \frac{1}{x}\right) - (x+s-1)\varepsilon,\ x \geq 1$, достигает наибольшего значения, где $[z]$ – наибольшее целое, не превосходящее $z$, $n$ – заданное число.

По данным примера число процессов $n_0 = \left[\sqrt{\frac{(r-1)T_\varepsilon^n}{(k+1)\varepsilon}}\right] + 1 = \left[\sqrt{\frac{40}{2}}\right] + 1 = [4{,}47] + 1 = 5$, что подтверждает ранг полученной оптимальной компоновки блоков.**Теорема.** Для того, чтобы эффективная одинаково распределенная система конкурирующих процессов в случае ограниченного параллелизма в асинхронном и втором синхронном режимах была оптимальной при заданных $p \geq 2,\ T_\varepsilon^n,\ \varepsilon > 0$, необходимо и достаточно, чтобы она была стационарной и число процессов $n_0$ в системе равнялось одному из чисел:

$$1)\ \left[\left[\sqrt{\frac{(p-1)T_\varepsilon^n}{k\varepsilon}}\right],\ \left[\sqrt{\frac{(p-1)T_\varepsilon^n}{k\varepsilon}}\right]+1\right] \cap [2,n],\ \text{при}\ s = kp,\ k > 1,$$

2) $\left[\left[\sqrt{\frac{(r-1)T_\varepsilon^n}{(k+1)\varepsilon}}\right],\ \left[\sqrt{\frac{(r-1)T_\varepsilon^n}{(k+1)\varepsilon}}\right]+1\right]\cap[2,n],$ при

$s=kp+r,\ k\geq 1,\ 1\leq r<p,$

в котором функция $\overline{\Delta}_\varepsilon(x)=(s-1)T^n\left(1-\frac{1}{x}\right)-(x+s-1)\varepsilon,\ x\geq 1,$ достигает наибольшего значения, где $[z]$ – наибольшее целое, не превосходящее $z$, $n$ – заданное число.

По данным примера число процессов

$$n_0=\left[\sqrt{\frac{(r-1)T_\varepsilon^n}{(k+1)\varepsilon}}\right]+1=\left[\sqrt{\frac{40}{2}}\right]+1=[4{,}47]+1=5,$$

что подтверждает ранг полученной оптимальной компоновки блоков.