EESTI NSV TEADUSTE AKADEEMIA TOIMETISED. 32. KÖIDE
FÜÜSIKA * MATEMAATIKA. 1983, NR. 2

ИЗВЕСТИЯ АКАДЕМИИ НАУК ЭСТОНСКОЙ ССР. ТОМ 32
ФИЗИКА * МАТЕМАТИКА. 1983, № 2

УДК 681.3.06

А. ТАУТС

РАСПАРАЛЛЕЛИВАНИЕ РЕКУРСИВНЫХ ВЫЧИСЛЕНИЙ

(Представил Н. Алумяэ)

Задаче распараллеливания рекурсии уделялось до сих пор мало внимания. В настоящей статье мы попытаемся дать некоторые указания в этом направлении. Основная идея состоит в том, что рекурсию можно начать еще до вычисления значения ее аргумента. Например, рекурсию

$$\varphi(y,0)=\varkappa(y),$$
$$\varphi(y,z+1)=\psi(y,z,\varphi(y,z)),$$

где $\varkappa$ и ψ — известные функции, можно начать, если известно значение для y, но еще не известно значение, которое примет z. Это значение можно вычислить параллельно с проведением рекурсии. Значительный эффект получается в случае, когда вышеуказанная рекурсия входит в качестве внутреннего цикла в следующую рекурсию

$$f(0)=a,$$
$$f(x+1)=h(\varphi(g(x),p(f(x),x)),f(x),x),$$

где h, g и p — известные функции. В случае, если нам надо вычислить, например, $f(n+1)$, то для проведения рекурсии надо вычислить $f(1)$, ..., $f(n)$, $f(n+1)$. Для этого нам понадобятся значения $\varphi(g(0),p(f(0),0))$, ..., $\varphi(g(n),p(f(n),n))$. Это значит, что вышеуказанную рекурсию для вычисления φ надо провести, выбирая в качестве y значения $g(0)$, ..., $g(n)$, и рекурсия по z дойдет в этих случаях до $p(f(0),0)$, ..., $p(f(n),n)$ соответственно. Последние вычисляются далее в ходе вычисления значений $f(i)$, $i=0,\ldots,n$, но все внутренние рекурсии можно начинать одновременно, выбирая в качестве y значения $g(0)$, ..., $g(n)$. Вычисляя одновременно $p(f(0),0)=p(a,0)$, узнаем, когда надо закончить первый внутренний цикл, после вычисления $p(f(1),1)$ узнаем конец второго внутреннего цикла и т. д. Выигрыш во времени получаем за счет того, что в тот момент, когда мы узнаем значение $p(f(i),i)$, большой участок рекурсии для вычисления $\varphi(g(i),p(f(i),i))$ уже пройден.

§ 1. Формализм для задания функций

Рассмотрим частично определенные функции, для которых в случае определенности значение вычисляется примитивно-рекурсивными средствами, в противном случае неопределенность обнаруживается тоже примитивно-рекурсивными средствами.

Исходными будут функции следующих двух типов:

1. Константные функции всяких арностей; они всегда определены независимо от определенности аргументов.

2. Функции $f(\bar{x}) = x_i$, т. е. значение i-го аргумента. Функция определена, если i-й аргумент определен.

Новые функции образуются следующим образом.

1. Подстановка: $g(\bar{x}) = f(f_1(\bar{x}), \ldots, f_n(\bar{x}))$. Функция $g(\bar{x})$ определена, если f определена для набора значений $f_1(\bar{x}), \ldots, f_n(\bar{x})$, среди которых могут быть и неопределенные.

2. Функция $g(\bar{x}, y) = \mu^-_{z \leqslant y} f(\bar{x}, z)$ — наименьшее из чисел $z \leqslant y$, для которых $f(\bar{x}, z) = 0$. Функция определена, если y определена и требуемое значение для z существует.

3. Функция $g(\bar{x}, y) = \mu^+_{z \leqslant y} f(\bar{x}, z)$ — наибольшее из чисел $z \leqslant y$, для которых $f(\bar{x}, z) = 0$. Определенность аналогична предыдущей.

4. Функция $g(\bar{x}, y) = \min_{z \leqslant y} f(\bar{x}, z)$ — минимальное значение функции. Функция $g(\bar{x}, y)$ определена, если y определена и $f(\bar{x}, z)$ определена хоть для некоторого $z \leqslant y$.

5. Функция $g(\bar{x}, y) = \max_{z \leqslant y} f(\bar{x}, z)$ — максимальное значение функции. Определенность аналогична предыдущей.

6. Функция $g(\bar{x}, y) = \sum_{z \leqslant y} f(\bar{x}, z)$. Она определена, если y определена и $f(\bar{x}, z)$ определена хоть для некоторого $z \leqslant y$. Неопределенные значения при суммировании опускаются.

7. Функция $g(\bar{x}, y) = \prod_{z \leqslant y} f(\bar{x}, z)$. Она определена, если y определена и $f(\bar{x}, z)$ определена для всех $z \leqslant y$.

8. Функция $h(\bar{x}, y)$, заданная неограниченной рекурсией $h(\bar{x}, 0) = f(\bar{x})$, $h(\bar{x}, y+1) = g(\bar{x}, y, h(\bar{x}, y))$. Она определена, если y и правая сторона равенства, определяющего значение $h(\bar{x}, y)$, определены.

9. Функция $h(\bar{x}, y)$, заданная ограниченной рекурсией $h(\bar{x}, 0) = f(\bar{x})$, $h(\bar{x}, y+1) = g(\bar{x}, y, h(\bar{x}, y))$, $h(\bar{x}, y) \leqslant r(\bar{x}, y)$. Она определена, если y, правая сторона соответствующего равенства и $r(\bar{x}, y)$ определены и выполнено неравенство $h(\bar{x}, y) \leqslant r(\bar{x}, y)$.

Ясно, что в результате добавления фиктивного аргумента u к исходным функциям некоторой операции, этот фиктивный аргумент добавляется к функции, полученной данной операцией. Поэтому фактически не требуется, чтобы функции, используемые некоторой операцией, имели все указанные аргументы, так как их можно фиктивно добавить.

Приведем некоторые примеры задания функций.

Функция $c(x)$, задаваемая неограниченной рекурсией $c(0) = 1$, $c(x+1) = 1$. Это постоянная функция 1, которая в случае неопределенности аргумента оказывается неопределенной.

Если имеются функции $f(\bar{x})$ и $g(\bar{x})$, то неограниченная рекурсия $h(\bar{x}, 0) = f(\bar{x})$, $h(\bar{x}, y+1) = g(\bar{x})$ задает функцию $h(\bar{x}, y)$, которая не определена, если y не определен, при $y = 0$ равна $f(\bar{x})$ и при $y \geqslant 1$ равна $g(\bar{x})$.

Используя теперь $\prod$ и подстановку, получим функцию $\prod_{y \leqslant 1} h(\bar{x}, y)$, которая равна $f(\bar{x}) \cdot g(\bar{x})$ и которая не определена, если хотя бы один из сомножителей не определен.

Функция $\sum_{y \leqslant 1} h(\bar{x}, y)$ равна $f(\bar{x}) + g(\bar{x})$. Она не определена, если оба слагаемых не определены. В случае неопределенности одного слагаемого сумма равна другому. Но так как функция $c(x)$ и операция умножения у нас есть, то мы можем $\sum_{y \leqslant 1} h(\bar{x}, y)$ умножить на $c(f(\bar{x})) \cdot c(g(\bar{x}))$ и получить сумму, для определенности которой требуется определенность обоих слагаемых.

Теперь ясно, что функцию $s(x) = x + 1$ можно получить, так как

функции x и 1 имеются. Так как в нашем распоряжении есть и остальные исходные функции и операции, необходимые для построения примитивно-рекурсивных функций, то все примитивно-рекурсивные функции содержатся в данной системе.

§ 2. Язык параллельных вычислений

Вышеописанный формализм указывает сущность каждой функции из рассматриваемого класса и операции для ее вычисления, но не говорит о моменте включения некоторого этапа в эти вычисления. С этой целью мы введем новый язык, используя следующие символы:

а) натуральные числа,

б) переменные отдельно для чисел и функций,

в) символы специальных операций μ^-, μ^+, min, max, Σ, Π,

г) символы порядка вычислений $*$, $|$, $\uparrow$,

д) вспомогательные символы — скобки, запятую, точку с запятой, двоеточие.

Определим терм, а также свободные и связанные переменные в нем.

1) Натуральные числа и числовые переменные — термы. Числовая переменная считается свободной.

2) Если f — n-местная функциональная переменная и $t_1, \ldots, t_n$ — термы, причем нет числовых переменных, содержащихся в t_i свободно и в t_j связано, то $f(t_1, \ldots, t_n)$ — терм. В нем каждая числовая переменная свободна или связана в зависимости от ее состояния в терме $t_1, \ldots. t_n$.

3) Если t_0 и t_1 — термы и нет числовых переменных, содержащихся в одном из них свободно и в другом связанно, а x — числовая переменная, которой в t_0 нет и которая не связана в t_1, то $\mu^-_{x\leqslant t_0} t_1$, $\mu^+_{x\leqslant t_0} t_1$, $\min_{x\leqslant t_0} t_1$, $\max_{x\leqslant t_0} t_1$, $\Sigma_{x\leqslant t_0} t_1$, $\Pi_{x\leqslant t_0} t_1$ — термы. В них x — связана, остальные числовые переменные свободны или связаны в зависимости от их вхождений в t_0 и t_1.

Определим процедуру, ее однозначность и многозначность, связанные и несвязанные числовые переменные, а также определенные и неопределенные функциональные переменные в ней.

1) Каждый терм есть однозначная процедура. В ней каждая числовая переменная свободна или связана в соответствии с определением ее состояния в терме. Все функциональные переменные не определены.

2) Если τ_1 и τ_2 — процедуры и нет числовых переменных, связанных в одной из них и свободных в другой, а функциональные переменные, существенно содержащиеся в обеих (см. ниже), не определены в обеих, то $\tau_1|\tau_2$ — многозначная процедура.

Свободность и связанность каждой числовой переменной, также как и определенность и неопределенность каждой функциональной переменной определяются ее состоянием в процедурах τ_1, τ_2.

3) Если τ_1 и τ_2 — процедуры и нет числовых переменных, связанных в одной из них и свободных в другой, а каждая функциональная переменная, существенно содержащаяся в обеих, либо не определена в обеих, либо определена только в τ_1, то $\tau_1 * \tau_2$ — однозначная или многозначная процедура в зависимости от вида τ_2. Каждая числовая переменная свободна или связана в зависимости от ее состояния в τ_1 и τ_2, а любая функциональная переменная определена, если она определена хотя бы в одной из τ_1 и τ_2.

4) Если τ_1 и τ_2 — процедуры, из них τ_2 — однозначная, $t_0, t_1, \ldots, t_{n-1}, t'$ и t'' — термы, нет числовых переменных, свободных в одной и связанных в другой из них, функциональные переменные, определенные в τ_2, не содержатся существенно в τ_1, t_0, t_1, $\ldots$, t_{n-1}, t' и t'', функциональные переменные, определенные в τ_1, не содержатся в $t_0, t_1, \ldots, t_{n-1}$ и t', а в τ_2 и t'' могут содержаться только неопределенно, x — числовая переменная, не содержащаяся в $t_0, t_1, \ldots, t_{n-1}$ и t', а в τ_1, τ_2 и t'' может содержаться только свободно, f — n-местная функциональная переменная, не содержащаяся в $t_0, t_1, \ldots, t_{n-1}, t', t''$ и τ_1, а в τ_2 может содержаться только неопределенно, то $(x \uparrow^{t_0} \tau_1)$, $(x \uparrow^{t_0} f(t_1, \ldots, t_{n-1}, x) : (t'; \tau_2))$, $(x \uparrow^{t_0} \tau_1 | f(t_1, \ldots, t_{n-1}, x) : (t'; \tau_2))$ и $(x \uparrow^{t_0} \tau_1 | f(t_1, \ldots, t_{n-1}, x) : (t'; \tau_2; t''))$ — многозначные процедуры. Переменная x в них связана, состояние остальных числовых переменных зависит от их состояния в $t_0, t_1, \ldots, t_{n-1}, t', t'', \tau_1$ и τ_2, переменная f определена, остальные функциональные переменные определены, если они определены в τ_1 или τ_2.

Так как каждая однозначная процедура является термом или имеет вид $\tau_1 * \tau_2$, где τ_2 — однозначная процедура, то индукцией определяется терм, который мы будем называть значением этой однозначной процедуры. Если процедура имеет вид терма, то она сама представляет свое значение, а значением процедуры $\tau_1 * \tau_2$ является значение τ_2.

Процедура называется определенной, если в ней нет неопределенных функциональных переменных. Процедура, как и терм, замкнуты, если в них нет свободных числовых переменных.

Если процедура имеет часть вида $x \uparrow^{t_0}$, то все вхождения, содержащиеся в этой части в составе t_0, называются несущественными вхождениями. Все остальные вхождения называются существенными.

Интерпретация описанного языка следующая. Символы $|$ и $*$ означают соответственно параллельное и последовательное применение процедур. $x \uparrow^{t_0} \tau_1$ означает применение τ_1 по всем значениям x до t_0 включительно, $f(t_1, \ldots, t_{n-1}, x) : (t'; \tau_2)$ и $f(t_1, \ldots, t_{n-1}, x) : (t'; \tau_2; t'')$ означают соответственно проведения неограниченной и ограниченной рекурсии, где t' есть исходное значение f при $x = 0$, τ_2 есть процедура перехода от x к $x + 1$, а t'' есть верхняя грань определяемой функции.

Однозначность и многозначность выражаются свойством процедуры вычислять одно конкретное значение или семейство значений.

§ 3. Синтез процедур параллельного вычисления

Рассмотрим, как перейти от определений, заданных в § 1, к записи § 2.

При переходе от некоторого определения к соответствующей процедуре будем предполагать, что имеются соответствующие процедуры для тех функций, на которые опирается данное определение.

Процедура, соответствующая определению некоторой функции, будет однозначной, определенной и содержащей в качестве свободных переменных только аргументы данной функции.

Мы имеем в виду, что в любой процедуре связанные числовые переменные и определенные функциональные переменные могут быть переименованы, если это окажется необходимым для допустимости использования некоторых процедур в построении новых процедур.

Исходным функциям поставим в соответствие процедуры, имеющие вид элементарных термов, т. е. натуральные числа и числовые переменные.

Пусть функция определена подстановкой вида $f(f_1(\bar{x}), \ldots, f_n(\bar{x}))$. Тогда запишем процедуру

$$[\tau_1(\bar{x})\,|\ldots|\,\tau_n(\bar{x})]\ast\tau(t_1(\bar{x}),\ \ldots,\ t_n(\bar{x})),$$

где $\tau_1, \ldots, \tau_n$ — процедуры функций $f_1, \ldots, f_n$ соответственно, τ — процедура функции f, а $t_1, \ldots, t_n$ — значения процедур $\tau_1, \ldots, \tau_n$ соответственно, которые поставлены в τ вместо аргументов функции f. При этом, если $t_i(\bar{x})$ имеет вид $\varphi(p_1, \ldots, p_k)$, где φ — какая-то функциональная переменная, то $t_i(\bar{x})$ в конце процедуры $\tau_i(\bar{x})$ опускается, т. е. вместо $\tau_i'(\bar{x})\ast t_1(\bar{x})$ пишут только $\tau_i'(\bar{x})$. $\tau_i(\bar{x})$, имеющие вид натуральных чисел или числовых переменных, вообще опускаются.

После этого совершается т. н. сдвиг уровня. Он состоит в следующем: если имеется процедура $\tau'\ast\tau''\ast\tau'''$ или $\tau'\ast(\tau''|\tau''')$, и τ'' не содержит существенного вхождения терма, процедура вычисления которого содержится в τ', то указанная запись заменяется на $(\tau'|\tau'')\ast\tau'''$. В данном случае сдвиг возможен, если в роли τ' будет $[\tau_1(\bar{x})|\ldots$ $\ldots|\tau_n(\bar{x})]$, а τ'' входит в состав $\tau(t_1(\bar{x}),\ \ldots,\ t_n(\bar{x}))$ и не содержит существенных вхождений термов $t_1(\bar{x}), \ldots, t_n(\bar{x})$. Мы будем предполагать, что сдвиг уровня осуществляется и в описываемых ниже процедурах, если это возможно. О характерных препятствиях сдвигу речь пойдет ниже.

Пусть функция имеет вид $\mu^-_{z\leqslant y}f(\bar{x},z)$, $\mu^+_{z\leqslant y}f(\bar{x},z)$, $\min_{z\leqslant y}f(\bar{x},z)$, $\max_{z\leqslant y}f(\bar{x},z)$, $\sum_{z\leqslant y}f(\bar{x},z)$ или $\prod_{z\leqslant y}f(\bar{x},z)$. В этом случае записываем процедуры

$$(z\uparrow^y\tau(\bar{x},z))\ast\mu^-_{z\leqslant y}t(\bar{x},z),\ (z\uparrow^y\tau(\bar{x},z))\ast\mu^+t(\bar{x},z),\ (z\uparrow^y\tau(\bar{x},z))\ast$$

$$\ast\min_{z\leqslant y}t(\bar{x},z),\ (z\uparrow^y\tau(\bar{x},z))\ast\max_{z\leqslant y}t(\bar{x},z),\ (z\uparrow^y\tau(\bar{x},z))\ast$$

$$\ast\sum_{z\leqslant y}t(\bar{x},z)$$

или $(z\uparrow^y\tau(\bar{x},z))\ast\prod_{z\leqslant y}t(\bar{x},z)$ соответственно, где τ — процедура для f и t — ее значение. При этом в конце процедуры $\tau(\bar{x},z)$ опускаем $t(\bar{x},z)$, если $t(\bar{x},z)$ имеет вид $\varphi(p_1, \ldots, p_k)$ при некоторой функциональной переменной φ. Если $\tau(\bar{x},z)$ есть натуральное число или числовая переменная, то $(z\uparrow^y\tau(\bar{x},z))$ вообще опускается. Здесь сдвиг уровня между $(z\uparrow^y\tau(\bar{x},z))$ и следующим за ним термом, содержащим $t(\bar{x},z)$, невозможен, так как $t(\bar{x},z)$ вычисляется процедурой $\tau(\bar{x},z)$. Но если $\tau(\bar{x},z)$ есть $\tau''\ast\tau'''$ или $(\tau'|\tau'')\ast\tau'''$, и τ'' не имеет существенных вхождений z, то $(z\uparrow^y\tau(\bar{x},z))$ заменяется на $\tau''\ast(z\uparrow^y\tau''')$ или на $\tau''\ast(z\uparrow^y(\tau'\ast\tau'''))$ соответственно. При этом, если в τ'' имеется $\uparrow^{t_0}$ и t_0 содержит z, то $\uparrow^{t_0}$ заменяется на $\uparrow^{\max_{z\leqslant y}t_0}$.

В случае, если τ вообще не содержит существенных вхождений z, то вместо $(z\uparrow^y\tau)$ пишем τ, заменяя в нем любое вхождение вида $\uparrow^{t'(z)}$ на $\uparrow^{\max_{z\leqslant y}t'(z)}$.

Пусть теперь функция $h(\bar{x},y)$ задана неограниченной или ограниченной рекурсией по схеме

$$h(\bar{x},0)=f(\bar{x}),\quad h(\bar{x},z+1)=g(\bar{x},z,h(\bar{x},z)),$$

а в случае ограниченной рекурсии верхняя грань для $h(\bar{x},y)$ задана функцией $r(\bar{x},y)$. Тогда запишем процедуру

$$\tau_1(\bar{x})\ast(z\uparrow^y\tau_2(\bar{x},z)\,|\,\bar{h}(\bar{x},z):(t_1(\bar{x});\tau_3(\bar{x},z,\bar{h}(\bar{x},z)),$$

$$t_2(\bar{x},z)))\ast\bar{h}(\bar{x},y),$$

где $\tau_1(\bar{x})$ и $t_1(\bar{x})$ — процедура функции $f(\bar{x})$ и ее значение, $\tau_2(\bar{x},z)$ и $t_2(\bar{x},z)$ — процедура функции $r(\bar{x},z)$ и ее значение, $\tau_3(\bar{x},z,n)$ —

процедура функции $g(\bar{x}, z, n)$ и $\bar{h}$ — некоторый не использованный до сих пор функциональный символ.

В случае неограниченной рекурсии $\tau_2(\bar{x}, z)$ и $t_2(\bar{x}, z)$ опускаются. Аналогично, в конце процедур $\tau_1(\bar{x})$ и $\tau_2(\bar{x})$ опускаем их значения, если они имеют вид $\varphi(p_1, \ldots, p_k)$. Если $\tau_1(\bar{x})$ или $\tau_2(\bar{x}, z)$ является натуральным числом или числовой переменной, то она вообще опускается.

Здесь сдвиг уровня между $\bar{h}(\bar{x}, y)$ и стоящей перед ней процедурой, определяющей $\bar{h}$, невозможен. Но если τ_3 имеет вид $\tau'' * \tau'''$ или $(\tau' | \tau'') * \tau'''$, и τ'' не имеет существенных вхождений $\bar{h}$, то τ_3 заменяем на τ''', соответственно $\tau' * \tau'''$, а τ_2 на $\tau_2 | \tau''$, а в случае отсутствия τ_2 пишем τ'' на ее место. В случае, если τ_3 вообще не содержит h, то $\bar{h}(\bar{x}, z) : (t_1(\bar{x}); \tau_3(\bar{x}, z, \bar{h}(\bar{x}, z)); t_2(\bar{x}, z))$ принимает вид $\bar{h}(\bar{x}, z) : (t_1(\bar{x}); t_3(\bar{x}, z, \bar{h}(\bar{x}, z)); t_2(\bar{x}, z))$, где t_3 — значение процедуры τ_3, а τ_3 присоединяется к τ_2 при помощи $|$ или ставится вместо τ_2 в случае его отсутствия. При этом значение τ_3 в конце снимают, если оно имеет вид $\varphi(p_1, \ldots, p_k)$, или τ_3 вообще опускают, если она является натуральным числом или числовой переменной.

Если теперь между $\uparrow^y$ и $\bar{h}$ оказывается некоторая процедура $\tau_0(\bar{x}, z)$, то, аналогично предыдущим случаям, устраняем из нее ту часть, которая не содержит существенных вхождений z, присоединяем ее с помощью символа $|$ к τ_1 и заменяем при этом каждое вхождение $\uparrow^{t_0(z)}$ на $\uparrow^{\max_{z \leqslant y} t_0(z)}$.

Полученные процедуры интерпретируются следующим образом. Ячейка предназначена для хранения в памяти некоторого замкнутого терма и только для него. Пока его значение не вычислено, ячейка остается пустой, причем при обращении к ней пустоту можно обнаружить. При $\tau_1 | \tau_2$ процедуры τ_1 и τ_2 включаются одновременно, при $\tau_1 * \tau_2$ процедура τ_2 включается после включения τ_1, но не обязательно после окончания вычисления τ_1. Если некоторый процессор нуждается в значении некоторого замкнутого терма, а это значение пока не вычислено, то процессор ждет до вычисления требуемого значения. Запись $x \uparrow^{t_0} \tau(x)$ означает, что до вычисления значения t_0 процедура τ при разных значениях x включается последовательно, а после вычисления значения t_0 — одновременно. Проведение рекурсии при $\bar{h}(t_1; \tau; t_2)$ остается последовательным и после определения верхней грани аргумента, если τ содержит символ функции $\bar{h}$.

Проиллюстрируем описанную технику примером. Возьмем пример из начала данной статьи, где для конкретности примем

$$\varkappa(y) \equiv 1,\ \psi(u, v, w) \equiv u \cdot (v+1) \cdot w,\ a = 0,\ p(u, v) \equiv u,\ g(x) \equiv x,$$
$$h(u, v, w) \equiv u + v + w.$$

Процедуры для сложения и умножения чисел u и v будут иметь следующий вид

$$(w \uparrow^1 \varrho(u, v, w) : (u; v)) * \Sigma_{w \leqslant 1} \varrho(u, v, w)$$

и

$$(w \uparrow^1 \varrho(u, v, w) : (u; v)) * \Pi_{w \leqslant 1} \varrho(u, v, w).$$

Обозначим эти процедуры сокращенно через $+(u, v)$ и $\times(u, v)$, а их значения — через $u + v$ и $u \cdot v$ соответственно. Тогда процедура для $\psi(u, v, w)$ примет вид $+(v, 1) * \times(u, v+1) * \times(u \cdot (v+1), w)$ и процедура для $h(u, v, w)$ — вид $+(u, v) * +(u+v, w)$. Процедурами для $\varkappa, a, p$ и g будут термы, выражающие их значения.

Процедура для функции $\varphi(y, w)$ получает вид

$$(z \uparrow^{w} \overline{\varphi}(y, z) : (1; +(z, 1) * \times (y, z+1) * \times (y \cdot (z+1), \overline{\varphi}(y, z)))) * \overline{\varphi}(y, w).$$

Преобразованием эта процедура превращается в следующую

$$(z \uparrow^{w} +(z, 1) * \times (y, z+1) \mid \overline{\varphi}(y, z) : (1; \times (y \cdot (z+1), \overline{\varphi}(y, z)))) * \overline{\varphi}(y, w),$$

так как $+(z, 1) * \times (y, z+1)$ не содержит $\overline{\varphi}$.

Теперь процедура для функции $f(x)$ принимает вид

$$(u \uparrow^{x} \bar{f}(u) : (0; (z \uparrow^{\bar{f}(u)} +(z, 1) * \times (u, z+1) \mid \overline{\varphi}(u, z) : (1; \times (u \cdot (z+1),$$

$$\overline{\varphi}(u, z)))) * +(\overline{\varphi}(u, \bar{f}(u)), \bar{f}(u)) * +(\overline{\varphi}(u, \bar{f}(u)) + \bar{f}(u), u))) * \bar{f}(x).$$

Учитывая, что

$$(z \uparrow^{\bar{f}(u)} +(z, 1) * \times (u, z+1) \mid \overline{\varphi}(u, z) : (1; \times (u \cdot (z+1), \overline{\varphi}(u, z))))$$

не содержит существенных вхождений $\bar{f}$, процедуре можно придать вид

$$(u \uparrow^{x} (z \uparrow^{\bar{f}(u)} +(z, 1) * \times (u, z+1) \mid \overline{\varphi}(u, z) : (1; \times (u \cdot (z+1), \overline{\varphi}(u, z)))) \mid$$

$$\bar{f}(u) : (0; +(\overline{\varphi}(u, \bar{f}(u)), \bar{f}(u)) * +(\overline{\varphi}(u, \bar{f}(u)) + \bar{f}(u), u))) * \bar{f}(x).$$

Институт кибернетики
Академии наук Эстонской ССР

Поступила в редакцию
28 мая 1982

A. TAUTS

REKURSIIVSETE ARVUTUSTE PARALLEELSUSTAMINE

Artiklis on esitatud rekursiivsete funktsioonide paralleelse arvutamise programmi keel ja sünteesi reeglid. Erilist tähelepanu on pööratud võimalusele alustada rekursiooni enne rekursioonimuutuja väärtuse väljaarvutamist.

A. TAUTS

DIE PARALLELISIERUNG DER REKURSIVEN BERECHNUNGEN

Im vorliegenden Artikel betrachtet man die primitivrekursiven Funktionen. Diese Funktionen kann man aus Konstanten und Variablen durch die Substitution, das Maximum und das Minimum, die Auswahl des größten und des kleinsten Arguments, das Summieren und das Multiplizieren, die unbegrenzte und begrenzte Rekursion konstruieren. Weiter gibt man eine Sprache, in der die Ordnung der parallelen Berechnungen solcher Funktionen ausdruckbar ist. Diese Sprache enthält Symbole für Konstanten, Variablen, Maximum, Minimum, Auswahlen, Summieren, Multiplizieren, für aufeinanderfolgende und parallele Berechnungen und für Berechnungen der Massive. Es werden Regeln gegeben, mit Hilfe deren man aus der Definition jeder Funktion einen Ausdruck ihrer Berechnung zur Synthese bringen und dann diesen Ausdruck so verändern kann, daß möglichst viele Berechnungen parallel durchgeführt werden können. Hauptsächlich wird es dadurch erreicht, daß man die Rekursion beginnt, bevor der Wert der Rekursionsvariablen ausgerechnet ist.