Транслируемост на схеми и програми

Иво Стратев

14 ноември 2020 г.

Транслируемост от стандартни схеми/програми към рекурсивни

Транслируемост от стандартна схема към рекурсивна схема

Разглеждаме следната стандартна схема

$$
\begin{aligned}
&\mathtt{input}(\mathsf{X});\ \mathtt{output}(\mathsf{Y})\\
&\quad 1 : \mathsf{Y} := \mathbf{c}\\
&\quad 2 : \mathsf{if}\ \mathbf{p}(\mathsf{X})\ \mathsf{then}\ \mathtt{goto}\ 6\ \mathsf{else}\ \mathtt{goto}\ 3\\
&\quad 3 : \mathsf{Y} := \mathbf{g}(\mathsf{X}, \mathsf{Y})\\
&\quad 4 : \mathsf{X} := \mathbf{f}(\mathsf{X})\\
&\quad 5 : \mathtt{goto}\ 2\\
&\quad 6 : \mathsf{stop}
\end{aligned}
$$

Прилагаме алгоритъма от **Теоремата на Маккарти**, за да транслираме горната стандартна схема до рекурсивна.

$$
\begin{aligned}
&\mathsf{F}_1(\mathsf{X}, \mathsf{X})\ \mathsf{where}\\
&\quad \mathsf{F}_1(\mathsf{X}, \mathsf{Y}) = \mathsf{F}_2(\mathsf{X}, \mathbf{c})\\
&\quad \mathsf{F}_2(\mathsf{X}, \mathsf{Y}) = \mathsf{if}\ \mathbf{p}(\mathsf{X})\ \mathsf{then}\ \mathsf{F}_6(\mathsf{X}, \mathsf{Y})\ \mathsf{else}\ \mathsf{F}_3(\mathsf{X}, \mathsf{Y})\\
&\quad \mathsf{F}_3(\mathsf{X}, \mathsf{Y}) = \mathsf{F}_4(\mathsf{X}, \mathbf{g}(\mathsf{X}, \mathsf{Y}))\\
&\quad \mathsf{F}_4(\mathsf{X}, \mathsf{Y}) = \mathsf{F}_5(\mathbf{f}(\mathsf{X}), \mathsf{Y})\\
&\quad \mathsf{F}_5(\mathsf{X}, \mathsf{Y}) = \mathsf{F}_2(\mathsf{X}, \mathsf{Y})\\
&\quad \mathsf{F}_6(\mathsf{X}, \mathsf{Y}) = \mathsf{Y}
\end{aligned}
$$

Забелязваме, че доста неща могат да бъдат оптимизирани. Например F_6 е селектираща функция, F_5 е преименуваща затова като първа стъпка ще ги inlineнем.

$$\begin{aligned}
&\mathsf{F}_1(\mathsf{X},\mathsf{X}) \text{ where}\\
&\quad \mathsf{F}_1(\mathsf{X},\mathsf{Y}) = \mathsf{F}_2(\mathsf{X},\mathbf{c})\\
&\quad \mathsf{F}_2(\mathsf{X},\mathsf{Y}) = \text{if } \mathbf{p}(\mathsf{X}) \text{ then } \mathsf{Y} \text{ else } \mathsf{F}_3(\mathsf{X},\mathsf{Y})\\
&\quad \mathsf{F}_3(\mathsf{X},\mathsf{Y}) = \mathsf{F}_4(\mathsf{X},\mathbf{g}(\mathsf{X},\mathsf{Y}))\\
&\quad \mathsf{F}_4(\mathsf{X},\mathsf{Y}) = \mathsf{F}_2(\mathbf{f}(\mathsf{X}),\mathsf{Y})\\
&\quad \mathsf{F}_5(\mathsf{X},\mathsf{Y}) = \mathsf{F}_2(\mathsf{X},\mathsf{Y})\\
&\quad \mathsf{F}_6(\mathsf{X},\mathsf{Y}) = \mathsf{Y}
\end{aligned}$$

Така достигаме до рекурсивната схема

$$\begin{aligned}
&\mathsf{F}_1(\mathsf{X},\mathsf{X}) \text{ where}\\
&\quad \mathsf{F}_1(\mathsf{X},\mathsf{Y}) = \mathsf{F}_2(\mathsf{X},\mathbf{c})\\
&\quad \mathsf{F}_2(\mathsf{X},\mathsf{Y}) = \text{if } \mathbf{p}(\mathsf{X}) \text{ then } \mathsf{Y} \text{ else } \mathsf{F}_3(\mathsf{X},\mathsf{Y})\\
&\quad \mathsf{F}_3(\mathsf{X},\mathsf{Y}) = \mathsf{F}_4(\mathsf{X},\mathbf{g}(\mathsf{X},\mathsf{Y}))\\
&\quad \mathsf{F}_4(\mathsf{X},\mathsf{Y}) = \mathsf{F}_2(\mathbf{f}(\mathsf{X}),\mathsf{Y})
\end{aligned}$$

След това започваме последователно inline-ване на функциите, които позволяват това. Започваме от F_4.

$$\begin{aligned}
&\mathsf{F}_1(\mathsf{X},\mathsf{X}) \text{ where}\\
&\quad \mathsf{F}_1(\mathsf{X},\mathsf{Y}) = \mathsf{F}_2(\mathsf{X},\mathbf{c})\\
&\quad \mathsf{F}_2(\mathsf{X},\mathsf{Y}) = \text{if } \mathbf{p}(\mathsf{X}) \text{ then } \mathsf{Y} \text{ else } \mathsf{F}_2(\mathbf{f}(\mathsf{X}),\mathbf{g}(\mathsf{X},\mathsf{Y}))\\
&\quad \mathsf{F}_3(\mathsf{X},\mathsf{Y}) = \mathsf{F}_2(\mathbf{f}(\mathsf{X}),\mathbf{g}(\mathsf{X},\mathsf{Y}))
\end{aligned}$$

Така достигаме до рекурсивната схема

$$\begin{aligned}
&\mathsf{F}_1(\mathsf{X},\mathsf{X}) \text{ where}\\
&\quad \mathsf{F}_1(\mathsf{X},\mathsf{Y}) = \mathsf{F}_2(\mathsf{X},\mathbf{c})\\
&\quad \mathsf{F}_2(\mathsf{X},\mathsf{Y}) = \text{if } \mathbf{p}(\mathsf{X}) \text{ then } \mathsf{Y} \text{ else } \mathsf{F}_2(\mathbf{f}(\mathsf{X}),\mathbf{g}(\mathsf{X},\mathsf{Y}))
\end{aligned}$$

Забелязваме, че F_1 изглежда някак излишна. Затова ще я премахнем със следната трансформация

$$\begin{aligned}
&\mathsf{F}_2(\mathsf{X},\mathbf{c}) \text{ where}\\
&\quad \mathsf{F}_1(\mathsf{X},\mathsf{Y}) = \mathsf{F}_2(\mathsf{X},\mathbf{c})\\
&\quad \mathsf{F}_2(\mathsf{X},\mathsf{Y}) = \text{if } \mathbf{p}(\mathsf{X}) \text{ then } \mathsf{Y} \text{ else } \mathsf{F}_2(\mathbf{f}(\mathsf{X}),\mathbf{g}(\mathsf{X},\mathsf{Y}))
\end{aligned}$$

Така получаваме рекурсивна схема от едно единствено уравнение.

$$\begin{aligned}
&\mathsf{F}_2(\mathsf{X},\mathbf{c}) \text{ where}\\
&\quad \mathsf{F}_2(\mathsf{X},\mathsf{Y}) = \text{if } \mathbf{p}(\mathsf{X}) \text{ then } \mathsf{Y} \text{ else } \mathsf{F}_2(\mathbf{f}(\mathsf{X}),\mathbf{g}(\mathsf{X},\mathsf{Y}))
\end{aligned}$$

Сега ако интерпретираме рекурсивната схема в типа данни (едносортната структура) $\langle \mathbb{N};\ 1;\ x \mapsto x \dot{-} 1,\ x, y \mapsto x.y;\ x \propto x = 0 \rangle$ получаваме рекурсивната програма

$$
\begin{aligned}
&\mathsf{F}_2(\mathsf{X}, \mathbf{1}) \text{ where} \\
&\quad \mathsf{F}_2(\mathsf{X}, \mathsf{Y}) = \text{if } \mathsf{X} = 0 \text{ then } \mathsf{Y} \text{ else } \mathsf{F}_2(\mathbf{X} \dot{-} 1, \mathbf{X}.\mathsf{Y})
\end{aligned}
$$

Транслируемост от стандартна програма към рекурсивна програма и определяне на семантики на опашкови функции

Разглеждаме следната стандартна схема

$$
\begin{aligned}
&\mathit{input}(\mathsf{X});\ \mathit{output}(\mathsf{Y}) \\
&\quad 1 : \mathsf{Y} := \mathbf{0} \\
&\quad 2 : \text{if } \mathsf{X} \leqslant \mathbf{1} \text{ then goto } 6 \text{ else goto } 3 \\
&\quad 3 : \mathsf{X} := \mathsf{X}\ \mathbf{div}\ \mathbf{2} \\
&\quad 4 : \mathsf{Y} := \mathsf{Y} + \mathbf{1} \\
&\quad 5 : \text{goto } 2 \\
&\quad 6 : \text{stop}
\end{aligned}
$$

Прилагаме алгоритъма от **Теоремата на Маккарти**, за да транслираме горната стандартна схема до рекурсивна.

$$
\begin{aligned}
&\mathsf{F}_1(\mathsf{X}, \mathsf{X}) \text{ where} \\
&\quad \mathsf{F}_1(\mathsf{X}, \mathsf{Y}) = \mathsf{F}_2(\mathsf{X}, \mathbf{0}) \\
&\quad \mathsf{F}_2(\mathsf{X}, \mathsf{Y}) = \text{if } \mathsf{X} \leqslant \mathbf{1} \text{ then } \mathsf{F}_6(\mathsf{X}, \mathsf{Y}) \text{ else } \mathsf{F}_3(\mathsf{X}, \mathsf{Y}) \\
&\quad \mathsf{F}_3(\mathsf{X}, \mathsf{Y}) = \mathsf{F}_4(\mathsf{X}\ \mathbf{div}\ \mathbf{2}, \mathsf{Y}) \\
&\quad \mathsf{F}_4(\mathsf{X}, \mathsf{Y}) = \mathsf{F}_5(\mathsf{X}, \mathsf{Y} + \mathbf{1}) \\
&\quad \mathsf{F}_5(\mathsf{X}, \mathsf{Y}) = \mathsf{F}_2(\mathsf{X}, \mathsf{Y}) \\
&\quad \mathsf{F}_6(\mathsf{X}, \mathsf{Y}) = \mathsf{Y}
\end{aligned}
$$

Правим първи пас на inline-ването.

$$
\begin{aligned}
&\mathsf{F}_2(\mathsf{X}, \mathbf{0}) \text{ where} \\
&\quad \mathsf{F}_1(\mathsf{X}, \mathsf{Y}) = \mathsf{F}_2(\mathsf{X}, \mathbf{0}) \\
&\quad \mathsf{F}_2(\mathsf{X}, \mathsf{Y}) = \text{if } \mathsf{X} \leqslant \mathbf{1} \text{ then } \mathsf{Y} \text{ else } \mathsf{F}_4(\mathsf{X}\ \mathbf{div}\ \mathbf{2}, \mathsf{Y}) \\
&\quad \mathsf{F}_3(\mathsf{X}, \mathsf{Y}) = \mathsf{F}_4(\mathsf{X}\ \mathbf{div}\ \mathbf{2}, \mathsf{Y}) \\
&\quad \mathsf{F}_4(\mathsf{X}, \mathsf{Y}) = \mathsf{F}_2(\mathsf{X}, \mathsf{Y} + \mathbf{1}) \\
&\quad \mathsf{F}_5(\mathsf{X}, \mathsf{Y}) = \mathsf{F}_2(\mathsf{X}, \mathsf{Y}) \\
&\quad \mathsf{F}_6(\mathsf{X}, \mathsf{Y}) = \mathsf{Y}
\end{aligned}
$$

Така достигаме до следната рекурсивна програма

$$\begin{aligned}
&\mathsf{F}_2(\mathsf{X}, \mathbf{0}) \text{ where}\\
&\quad \mathsf{F}_2(\mathsf{X}, \mathsf{Y}) = \text{if } \mathsf{X} \leqslant \mathbf{1} \text{ then } \mathsf{Y} \text{ else } \mathsf{F}_4(\mathsf{X}\ \mathbf{div}\ \mathbf{2}, \mathsf{Y})\\
&\quad \mathsf{F}_4(\mathsf{X}, \mathsf{Y}) = \mathsf{F}_2(\mathsf{X}, \mathsf{Y} + \mathbf{1})
\end{aligned}$$

Сега ясно се вижда, че остава F_4 да бъде inline-ната и така получаваме рекурсивната програма

$$\begin{aligned}
&\mathsf{F}_2(\mathsf{X}, \mathbf{0}) \text{ where}\\
&\quad \mathsf{F}_2(\mathsf{X}, \mathsf{Y}) = \text{if } \mathsf{X} \leqslant \mathbf{1} \text{ then } \mathsf{Y} \text{ else } \mathsf{F}_2(\mathsf{X}\ \mathbf{div}\ \mathbf{2}, \mathsf{Y} + \mathbf{1})
\end{aligned}$$

Сравнително лесно се забелязва, че денотационната семантика по стойност на F_2 е функцията f_2 дефинирана посредством правилото

$$f_2(x, y) \simeq y + \mathsf{Log}(x),$$

където Log е фукцията дефинирана по средством правилото

$$\mathsf{Log}(x) \simeq \begin{cases} 0 & , \ x \leqslant 1 \\ \lfloor \log_2(X) \rfloor & , \ x > 1 \end{cases}$$

Така денотационната семантика по стойност на програмата лесно се съобразява, че е Log.

Сега ако се върнем на рекурсивната програма, от която тръгнахме

$$\begin{aligned}
&\mathsf{F}_1(\mathsf{X}, \mathsf{XY}) \text{ where}\\
&\quad \mathsf{F}_1(\mathsf{X}, \mathsf{Y}) = \mathsf{F}_2(\mathsf{X}, \mathbf{0})\\
&\quad \mathsf{F}_2(\mathsf{X}, \mathsf{Y}) = \text{if } \mathsf{X} \leqslant \mathbf{1} \text{ then } \mathsf{F}_6(\mathsf{X}, \mathsf{Y}) \text{ else } \mathsf{F}_3(\mathsf{X}, \mathsf{Y})\\
&\quad \mathsf{F}_3(\mathsf{X}, \mathsf{Y}) = \mathsf{F}_4(\mathsf{X}\ \mathbf{div}\ \mathbf{2}, \mathsf{Y})\\
&\quad \mathsf{F}_4(\mathsf{X}, \mathsf{Y}) = \mathsf{F}_5(\mathsf{X}, \mathsf{Y} + \mathbf{1})\\
&\quad \mathsf{F}_5(\mathsf{X}, \mathsf{Y}) = \mathsf{F}_2(\mathsf{X}, \mathsf{Y})\\
&\quad \mathsf{F}_6(\mathsf{X}, \mathsf{Y}) = \mathsf{Y}
\end{aligned}$$

Имайки семантиката на функцията, определена от F_2, можем да намерим денотационната семантика по стойност на всяка една от шестте опашкови функции. Нека семантиките са $f_1, f_2, \ldots, f_6$. Тогава лесно се съобразват семантиките на f_1, f_2, f_5, f_6 и те са

$$\begin{aligned}
f_1(x, y) &\simeq \mathsf{Log}(x)\\
f_2(x, y) &\simeq y + \mathsf{Log}(x)\\
f_5(x, y) &\simeq y + \mathsf{Log}(x)\\
f_6(x, y) &\simeq y
\end{aligned}$$

Откъдето вече лесно се съобразяват и f_4 и f_3.

$$f_4(x, y) \simeq y + 1 + \mathsf{Log}(x)$$
$$f_3(x, y) \simeq y + 1 + \mathsf{Log}(x \ \mathbf{div}\ 2) \simeq y + \mathsf{Log}(x)$$

Свеждане на опашкова рекурсия до итерация

Tail call рекурсия

Разглеждаме следната двуместна частична функция над естествени числа

$$f(x, y) \simeq \begin{cases} \neg! & , \ y = 0 \\ x \ \mathbf{mod}\ y & , \ y > 0 \end{cases}$$

Сравнително лесно можем да се досетим за рекурсивна програма, на която тя е семантика. За целта ни трябват само две съображения. Първо, недефинираност лесно се моделира със зацикляне и $(Q + 1).Y + K \ \mathbf{mod}\ Y$ очевидно е K, ако $K < Y$, но също така е $Q.Y + K \ \mathbf{mod}\ Y$, което е $((Q + 1).Y + K) - Y \ \mathbf{mod}\ Y$. Така достигаме до следната рекурсивна програма

$$\begin{aligned} &F(X, Y) \text{ where} \\ &\qquad F(X, Y) = \text{if } Y = \mathbf{0} \text{ then } F(X, Y) \text{ else if } X < Y \text{ then } X \text{ else } F(X \dot{-} Y, Y) \end{aligned}$$

Добре е да забележим, че е излишно да повтаряме всеки път проверката $Y = 0$. За да я избегнем, разделяме програмата на две фукции.

$$\begin{aligned} &F_1(X, Y) \text{ where} \\ &\qquad F_1(X, Y) = \text{if } Y = \mathbf{0} \text{ then } F_1(X, Y) \text{ else } F_2(X, Y) \\ &\qquad F_2(X, Y) = \text{if } X < Y \text{ then } X \text{ else } F_2(X \dot{-} Y, Y) \end{aligned}$$

Сега лесно се забелязва, че самата F_2 също пресмята f. Също така обръщенията към F_2 в $\mathtt{else}$ клона и към F_1 и F_2 в тялото на F_1 са recursive tail call. Резултатът Y в основният клон на $\mathtt{if}$-а за F_2 също можем да си мислим, че е tail call към селектираща функция. Така понеже резултатът във всеки клон е tail call, то гарантирано рекурсивната програма може да бъде сведена до итеративна (стандартна) програма.

Първо да забележим, че променливата, която е резултатна, е X. След това да направим обратното на това, което правихме до сега, без да попълваме етикетите на скоковете (jump-овете).

$$
\begin{aligned}
&\mathtt{input}(X, Y);\ \mathtt{output}(X)\\
&\quad 1\ :\ \text{if } Y = \mathbf{0} \text{ then } \mathtt{goto}\ \sqcup\ \text{ else } \mathtt{goto}\ \sqcup\\
&\quad 2\ :\ \text{if } X < Y \text{ then } \mathtt{goto}\ \sqcup\ \text{ else } \mathtt{goto}\ \sqcup\\
&\quad 3\ :\ X := X \dot{-} Y\\
&\quad 5\ :\ \mathtt{goto}\ \sqcup\\
&\quad 6\ :\ \text{stop}
\end{aligned}
$$

Етикетите на скоковете би трябвало вече да са ясни, така че направо ги попълваме и получаваме програмата

$$
\begin{aligned}
&\mathtt{input}(X, Y);\ \mathtt{output}(X)\\
&\quad 1\ :\ \text{if } Y = \mathbf{0} \text{ then } \mathtt{goto}\ 1 \text{ else } \mathtt{goto}\ 2\\
&\quad 2\ :\ \text{if } X < Y \text{ then } \mathtt{goto}\ 6 \text{ else } \mathtt{goto}\ 3\\
&\quad 3\ :\ X := X \dot{-} Y\\
&\quad 5\ :\ \mathtt{goto}\ 2\\
&\quad 6\ :\ \text{stop}
\end{aligned}
$$

Последен пример

Разглеждаме следната рекурсивна схема

$$
\begin{aligned}
&F(X) \text{ where}\\
F(X) = \text{if } \mathbf{isBaseCase}(X) \text{ then } \mathbf{base}(X) \text{ else } &\mathbf{cons}(X,\ F(\mathbf{left}(X)),\ F(\mathbf{right}(X)))
\end{aligned}
$$

Въпрос: Ако интерпретираме тази схема в типа данни на наредените двоичните крайни коренови дървета с върхове финитно подмножество на естествените числа, където интерпретацията на **isBaseCase** е проверка дали дървото е празно (множеството от върхове е празно), **base** се интерпретира като идентитет, а **left** и **right** като ляво поддърво и дясно поддърво и **cons** връща дърво с корен - корена на X, увеличен с единица, и поддървета - другите два аргумента. Тогава вярно ли е, че рекурсивната програма може да бъде транслирана до стандартна?

Сега ще покажем, че ако схемата бъде интерпретирана в един конкретен тип данни обаче може да бъде транслирана. Разглеждаме типа дани $\langle \mathbb{N};\ b,\ l,\ r,\ t;\ s\rangle$, където $s = \{0, 1\}$, $t(n, k, u) = n + k.u$, $l(n) = n \dot{-} 2$, $r(n) = n \dot{-} 1$ и

$$
b(x) = \begin{cases} 2 & , x = 0 \\ 3 & , x = 1 \\ x & , x > 1 \end{cases}
$$

Така получаваме рекурсивната програма.

$$\begin{aligned} &\mathsf{F(X)\ where} \\ \mathsf{F(X)} &= \mathsf{if\ X \leqslant 1\ then\ \mathbf{b}(X)\ else\ X + F(X \dot{-} 2).F(X \dot{-} 1)} \end{aligned}$$

Семантиката на тази програма реално е една рекурентна изброима редица с елементи естествени числа, съотвестваща на рекурентното уравнение.

$$\begin{aligned} a_0 &= 2 \\ a_1 &= 3 \\ a_{n+2} &= n + 2 + a_n.a_{n+1} \end{aligned}$$

Рекуретната връзка е нелинейна, но рекурентното уравнение, задаващо редицата, е с крайна история. Затова можем да напишем почти екивалетна акумулаторна рекурсивна програма, която да транслираме до стандартна. Почти еквивалетна, защото трябва да обогатим типа данни (структурата) с три константи и една операция.

$$\begin{aligned} &\mathsf{F(X)\ where} \\ \mathsf{F(X)} &= \mathsf{if\ X \leqslant 1\ then\ \mathbf{b}(X)\ else\ G(X,\ 2,\ \mathbf{b}(0),\ \mathbf{b}(1))} \\ \mathsf{G(X, U, Y, Z)} &= \mathsf{if\ X \leqslant 1\ then\ Z\ else\ G(X \dot{-} 1,\ U + 1,\ Z,\ U + Y.Z)} \end{aligned}$$

При транслирането ще имаме нужда от още една променлива, защото иначе не бихме могли да извършим транслацията. Допълнителната променлива ще е T. Отново първо пишем код без етикети на скоковете, след това ги съобразяваме.

$$\begin{aligned} &\mathtt{input}(\mathsf{X});\ \mathtt{output}(\mathsf{Z}) \\ &\quad 1\ :\ \mathsf{if\ X \leqslant \mathbf{1}\ then}\ \mathtt{goto}\ \sqcup\ \ \mathsf{else}\ \mathtt{goto}\ \sqcup \\ &\quad 2\ :\ \mathsf{if\ X \leqslant \mathbf{1}\ then}\ \mathtt{goto}\ \sqcup\ \ \mathsf{else}\ \mathtt{goto}\ \sqcup \\ &\quad 3\ :\ \mathsf{Z} := \mathbf{b}(\mathsf{X}) \\ &\quad 4\ :\ \mathtt{goto}\ \sqcup \\ &\quad 5\ :\ \mathsf{U} := \mathbf{2} \\ &\quad 6\ :\ \mathsf{Y} := \mathbf{b}(\mathbf{0}) \\ &\quad 7\ :\ \mathsf{Z} := \mathbf{b}(\mathbf{1}) \\ &\quad 8\ :\ \mathtt{goto}\ \sqcup \\ &\quad 9\ :\ \mathsf{T} := \mathsf{U} + \mathsf{Y.Z} \\ &\quad 10\ :\ \mathsf{X} := \mathsf{X} \dot{-} 1 \\ &\quad 11\ :\ \mathsf{U} := \mathsf{U} + 1 \\ &\quad 12\ :\ \mathsf{Y} := \mathsf{Z} \\ &\quad 13\ :\ \mathsf{Z} := \mathsf{T} \\ &\quad 14\ :\ \mathtt{goto}\ \sqcup \\ &\quad 15\ :\ \mathsf{stop} \end{aligned}$$

Като сложим етикети на скоковете, получаваме програмата.

$$
\begin{array}{l}
\mathrm{input}(\mathsf{X});\ \mathrm{output}(\mathsf{Z})\\
\quad \mathsf{1}\ :\ \mathsf{if}\ \mathsf{X} \leqslant \mathbf{1}\ \mathsf{then\ goto}\ 3\ \mathsf{else\ goto}\ 5\\
\quad \mathsf{2}\ :\ \mathsf{if}\ \mathsf{X} \leqslant \mathbf{1}\ \mathsf{then\ goto}\ 15\ \mathsf{else\ goto}\ 9\\
\quad \mathsf{3}\ :\ \mathsf{Z} := \mathbf{b}(\mathsf{X})\\
\quad \mathsf{4}\ :\ \mathsf{goto}\ 15\\
\quad \mathsf{5}\ :\ \mathsf{U} := \mathbf{2}\\
\quad \mathsf{6}\ :\ \mathsf{Y} := \mathbf{b}(\mathbf{0})\\
\quad \mathsf{7}\ :\ \mathsf{Z} := \mathbf{b}(\mathbf{1})\\
\quad \mathsf{8}\ :\ \mathsf{goto}\ 2\\
\quad \mathsf{9}\ :\ \mathsf{T} := \mathsf{U} + \mathsf{Y}.\mathsf{Z}\\
\quad \mathsf{10}\ :\ \mathsf{X} := \mathsf{X} \dot{-} \mathsf{1}\\
\quad \mathsf{11}\ :\ \mathsf{U} := \mathsf{U} + \mathsf{1}\\
\quad \mathsf{12}\ :\ \mathsf{Y} := \mathsf{Z}\\
\quad \mathsf{13}\ :\ \mathsf{Z} := \mathsf{T}\\
\quad \mathsf{14}\ :\ \mathsf{goto}\ 2\\
\quad \mathsf{15}\ :\ \mathsf{stop}
\end{array}
$$

Сега ако влезем в ролята на компилатор, който оптимизира инструкциите, така че да са близки при повторение, за да се съберат в кеша за инструкции бихме пренаредили инструкциите по следния начин

$$
\begin{array}{l}
\mathrm{input}(\mathsf{X});\ \mathrm{output}(\mathsf{Z})\\
\quad \mathsf{1}\ :\ \mathsf{if}\ \mathsf{X} \leqslant \mathbf{1}\ \mathsf{then\ goto}\ 2\ \mathsf{else\ goto}\ 4\\
\quad \mathsf{2}\ :\ \mathsf{Z} := \mathbf{b}(\mathsf{X})\\
\quad \mathsf{3}\ :\ \mathsf{goto}\ 14\\
\quad \mathsf{4}\ :\ \mathsf{U} := \mathbf{2}\\
\quad \mathsf{5}\ :\ \mathsf{Y} := \mathbf{b}(\mathbf{0})\\
\quad \mathsf{6}\ :\ \mathsf{Z} := \mathbf{b}(\mathbf{1})\\
\quad \mathsf{7}\ :\ \mathsf{if}\ \mathsf{X} \leqslant \mathbf{1}\ \mathsf{then\ goto}\ 14\ \mathsf{else\ goto}\ 8\\
\quad \mathsf{8}\ :\ \mathsf{T} := \mathsf{U} + \mathsf{Y}.\mathsf{Z}\\
\quad \mathsf{9}\ :\ \mathsf{X} := \mathsf{X} \dot{-} \mathsf{1}\\
\quad \mathsf{10}\ :\ \mathsf{U} := \mathsf{U} + \mathsf{1}\\
\quad \mathsf{11}\ :\ \mathsf{Y} := \mathsf{Z}\\
\quad \mathsf{12}\ :\ \mathsf{Z} := \mathsf{T}\\
\quad \mathsf{13}\ :\ \mathsf{goto}\ 7\\
\quad \mathsf{14}\ :\ \mathsf{stop}
\end{array}
$$