

CS 5000: Lecture 36

Vladimir Kulyukin

Department of Computer Science

Utah State University

Outline

- Construction of the Universal Program U_n .
- A Computational Theory of Simulation

Universal Program U_n

Review: U_n Construction So Far

$Z \leftarrow X_{n+1} + 1$ // Get the source number.

$S \leftarrow \prod_{i=1}^n (p_{2i})^{X_i}$ // Encode the initial state.

$K \leftarrow 1$ // Initialize instruction counter.

[C] IF $K = Lt(Z) + 1 \vee K = 0$ GOTO E // Check termination

$U \leftarrow r((Z)_K)$ // Get $\langle b, c \rangle$ of K - th instruction.

$P \leftarrow p_{r(U)+1}$ // Get the prime corresponding variable # $c + 1$.

IF $l(U) = 0$ GOTO N // If $V \leftarrow V$, go to NOTHING.

IF $l(U) = 1$ GOTO A // If $V \leftarrow V + 1$, go to ADD.

IF $\neg(P | S)$ GOTO N // If $V = 0$, go to NOTHING.

IF $l(U) = 2$ GOTO M // If $V \leftarrow V - 1$, go to MINUS.

Review: Coding Conditional Dispatch

IF K - th instruction is of the form IF $V \neq 0$ GOTO L ,
then $b = \#(L) + 2$.

Review: U_n Construction

IF $l(U) > 2$ and $P \mid S$, then the current instruction is of the form

IF $V \neq 0$ GOTO L and $\#(L) = l(U) - 2$.

Review: U_n Construction

$$K \leftarrow \min_{i \leq Lt(Z)} [l((Z)_i) + 2 = l(U)]$$

GOTO C

If there is no such label, $K = 0$.

U_n Construction: Part 5

// Doing Nothing

$[N]$ $K \leftarrow K + 1$ // Increment instruction counter.

GOTO C // Go back C .

U_n Construction: Part 5

// Subtraction

$$[M]S \leftarrow \lfloor S/P \rfloor$$

For example, $S = 63 = ([0,2,0,1])$.

Suppose $(Z)_1 = \#(X \leftarrow X - 1)$. X is the same as $X1$ and

$X1$ is the second variable in our variable sequence. Thus,

$P = p_2 = 3$. Then the new state S' , after the subtraction is executed, is $S' \leftarrow \lfloor S/P \rfloor = \lfloor 63/3 \rfloor = 21 = 3 \cdot 7 = ([0,1,0,1])$.

U_n Construction: Part 5

// Addition

$$[A] S \leftarrow S \cdot P$$

For example, suppose $S = 63 = [0,2,0,1]$. Suppose $K = i$.
Suppose $(Z)_i = \#(X_2 \leftarrow X_{2+1})$. X_2 is the 4 - th variable
in our sequence. Thus, $P = p_4$. Then the new state
 $S' = [0,2,0,2] = 63 \cdot 7 = S \cdot P$.

U_n Construction: Part 5

$[M] S \leftarrow \lfloor S / P \rfloor$ // Subtract.

GOTO N

$[A] S \leftarrow S \cdot P$ // Add.

$[N] K \leftarrow K + 1$ // Increment instruction counter.

GOTO C // Back to C.

$[E] Y \leftarrow (S)_1$ // Set the value of the output variable.

U_n Construction: Final Cut

$$Z \leftarrow X_{n+1} + 1$$

$$S \leftarrow \prod_{i=1}^n (p_{2i})^{X_i}$$

$$K \leftarrow 1$$

[C] IF $K = Lt(Z) + 1 \vee K = 0$ GOTO E

$$U \leftarrow r((Z)_K)$$

$$P \leftarrow p_{r(U)+1}$$

IF $l(U) = 0$ GOTO N

IF $l(U) = 1$ GOTO A

IF $\neg(P | S)$ GOTO N

IF $l(U) = 2$ GOTO M

$$K \leftarrow \min_{i \leq Lt(Z)} [l((Z)_i) + 2 = l(U)]$$

GOTO C

$$[M] S \leftarrow \lfloor S / P \rfloor$$

GOTO N

$$[A] S \leftarrow S \cdot P$$

$$[N] K \leftarrow K + 1$$

GOTO C

$$[E] Y \leftarrow (S)_1$$

Remember What We Proved?

Theorem 3.1 (Universality Theorem) : For each $n > 0$, the function $\Phi^{(n)}(x_1, \dots, x_n, y)$ is partially computable.

A Simple Example

[B] $Y \leftarrow Y$

$Y \leftarrow Y$

$Y \leftarrow Y+1$

The number of the program = 199.

The program's source = $199 + 1 = 200$

A Simple Example

- $200 = [3, 0, 2]$
- $3 = \langle 2, \langle 0, 0 \rangle \rangle$
- $0 = \langle 0, \langle 0, 0 \rangle \rangle$
- $2 = \langle 0, \langle 1, 0 \rangle \rangle$

A Simple Example

Here are the instructions that we must execute:

1.<2, <0, 0>>

2.<0, <0, 0>>

3.<0, <1, 0>>

Another Simple Example

- $X \leftarrow X + 1$
- $X \leftarrow X + 1$

The number of this source code is [10, 10],
so we execute these compiled instructions:

1. $\langle 0, \langle 1, 1 \rangle \rangle$
2. $\langle 0, \langle 1, 1 \rangle \rangle$

Remember These Equalities

$$1. \Phi_y^{(n)}(x_1, \dots, x_n) = \Phi^{(n)}(x_1, \dots, x_n, y).$$

$$2. \Phi^{(n)}(x_1, \dots, x_n, y) = \Psi_P^{(n)}(x_1, \dots, x_n).$$

$$3. y = \#(P).$$

Enumeration of PC Functions

Is it possible to enumerate all partially computable functions of n variables?

Enumeration of PC Functions

Yes, it is. Here are the enumerations :

$$n = 1 : \Phi^{(1)}(x_1, 0), \Phi^{(1)}(x_1, 1), \Phi^{(1)}(x_1, 2), \dots$$

$$n = 2 : \Phi^{(1)}(x_1, x_2, 0), \Phi^{(1)}(x_1, x_2, 1), \Phi^{(1)}(x_1, x_2, 2), \dots$$

$$n = 3 : \Phi^{(1)}(x_1, x_2, x_3, 0), \Phi^{(1)}(x_1, x_2, x_3, 1), \Phi^{(1)}(x_1, x_2, x_3, 2), \dots$$

...

$$n = n : \Phi^{(n)}(x_1, \dots, x_n, 0), \Phi^{(n)}(x_1, \dots, x_n, 1), \Phi^{(n)}(x_1, \dots, x_n, 2), \dots$$

Review: Snapshots

$s = (i, \sigma), 1 \leq i \leq n + 1.$

i is the number of the instruction to be executed.

σ is a state of the program.

s is terminal if $i = n + 1.$

A Computational Theory of Simulation

Step-Counter Predicate

$$STP^{(n)}(x_1, \dots, x_n, y, t)$$

1. Program whose number is y halts on inputs x_1, \dots, x_n on t or fewer steps.
2. There is a computation of program number y whose length $\leq t + 1$, beginning with inputs x_1, \dots, x_n .

Recommended Reading

- Section 4.3.