Random Access Machines

An array of **registers** each of which can contain an arbitrarily large integer.

Instruction set: READ, STORE, LOAD, ADD, JUMP, JZERO, HALF, HALT, ...

Syntax: Instruction [operand]

A program counter.

Three modes of addressing: direct, register direct, register indirect.

A RAM program consists of a finite sequence of such instructions.

Input is a finite sequence of integers (in binary) contained in input array of input registers.

Output is contained in a special register (accumulator).

Assume that all arithmetic operation of RAM takes a single step.

TM can simulate RAM

Let Π be a RAM program and let D be a set of finite sequences of integers.

Let ϕ be a function from D to integers.

We say that Π computes ϕ if, for any $I \in D$,

the program, on input I, halts with $\phi(I)$ in the accumulator.

Let the binary representation of $I = (i_1, i_2, \ldots, i_n)$ denoted by $b(I)$ be the string $b(i_1); \ldots; b(i_n)$.

TM M computes ϕ if, for any $I \in D$, $M(b(I)) = b(\phi(I))$.

Theorem 1. *If a RAM program Π computes a function ϕ in time $f(n)$, then there is a 7-string TM M which computes ϕ in time $O(f(n)^3)$.*

Proof

1st string: Input, never overwritten.

2nd string: Register contents – sequence of (separated) strings of the form $b(i) : b(r_i)$.

3rd string: current value of program counter

4th string: current register address

5th string: operand 1

6th string: operand 2

7th string: result/output

The states of M are subdivided into m groups, where m is the number of instructions in Π; each group implements one instruction.

Time Complexity

Lemma 1. *After tth step of a RAM program computation on input I, the contents of any register have length at most $t + \ell(I) + \ell(B)$.*

Proof: By induction on t.

Easy to see that it holds for operations such as LOAD, STORE, READ etc.

Consider an arithmetic instruction e.g., ADD.

The length of the result is $\leq 1+t-1+\ell(I)+\ell(B)$.

Simulating each instruction of Π takes $O((f(n))^2$ time where n is the length of the input.

Decoding current instruction and the constants it contains can be done in constant time.

Fetching the value of the registers involved takes $O((f(n))^2)$ time:

(the second string contains $O(f(n))$ pairs each of length $O(f(n))$).

The computation involves arithmetic functions on integers of length $O(f(n)$ and can be done in $O(f(n))$.

Nondeterministic Turing Machines (NTM)

A NTM $N = (K, \Sigma, \Delta, s)$ is like a ordinary TM except that instead of a transition function we have a transition **relation** Δ.

$\Delta \subset (K \times \Sigma) \times [(K \cup \{h, yes, no\}) \times \Sigma \times \{\leftarrow, \rightarrow, -\}]$.

Many (or none) next steps for each state-symbol combination.

"yields" is a relation.

Definition 1. *We say that a NTM N* **decides** *a language L if for any $x \in \Sigma^*$, the following is true: $x \in L$ iff $(s, \triangleright, x) \rightarrow^{N^*} (yes, w, u)$ for some strings w and u.*

N decides L in time $f(n)$ if N decides L, and for any $x \in \Sigma^*$, if $(s, \triangleright, x) \rightarrow^{N^k} (q, w, u)$, then $k \leq f(|x|)$.