INF580 – Large-scale Mathematical Programming

TD2 (hard version) — Computability and MP

Leo Liberti

CNRS LIX, Ecole Polytechnique, France

Marvin Minsky's register machine (MRM)

- MRM is a qudruplet (R, N, S, c)
- $R = (R_1, R_2, \ldots)$: infinite sequence of registers
- $\forall i \in \mathbb{N}$, each R_i contains an integer
- $N = \{0, \ldots, n\}$ is a set of states
  $N^+ = N \smallsetminus \{0\}$
- $S : N^+ \to \mathbb{N} \times \{0, 1\} \times N \times N$ is a program
- c holds the current instruction index

MRM instructions

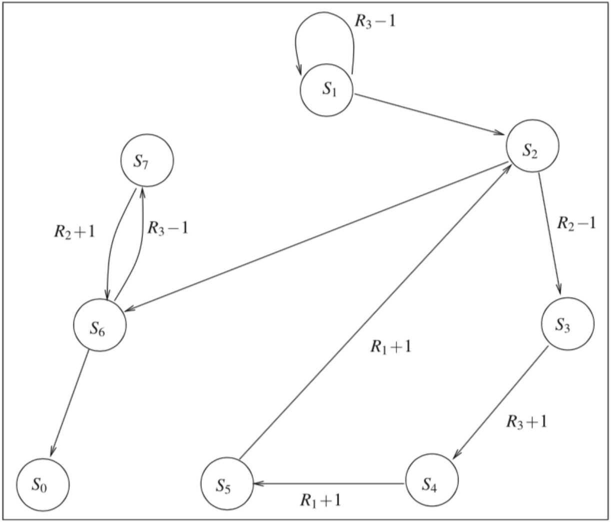
- Each instruction $S(i)$ (for $i \in N^+$) of a MRM program S is a quadruplet (j, b, k, ℓ)
- If $S(i) = (j, b, k, \ell)$ then $S(i)$ is an instruction of type $b \in \{0, 1\}$
  - if $b = 0$ then $R_j \leftarrow R_j + 1$ and $c \leftarrow k$
  - if $b = 1$ and $R_j > 0$ then $R_j \leftarrow R_j - 1$ and $c \leftarrow k$
  - if $b = 1$ and $R_j = 0$ then $c \leftarrow \ell$
- If $c = 0$ then MRM terminates
- If $b = 0$ then ℓ is unused

MRM example [Johnstone 87]

Algorithm:
$R_1 \leftarrow R_1 + 2R_2$

$$
\begin{aligned}
S_1 &= (3,1,1,2) \\
S_2 &= (2,1,3,6) \\
S_3 &= (3,0,4,0) \\
S_4 &= (1,0,5,0) \\
S_5 &= (1,0,2,0) \\
S_6 &= (3,1,7,0) \\
S_7 &= (2,0,6,0).
\end{aligned}
$$

```
while (R3 > 0)
      R3--;
while (R2 > 0) {
      R2--; R3++;
      R1++; R1++;
}
while (R3 > 0) {
      R3--;
      R2++;
}
```



Minsky's theorem

The MRM is a UTM

Proof: simulate a UTM using the MRM

Exercises

1. Execute "by hand" Johnstone's MRM example for inputs (R_1, R_2) in the set $\{(1,1),(2,1)\}$; make sure you obtain the correct output in R_1
2. Write a MRM program isfactorof(a, b) which tests if $a|b$
3. Devise a MP formulation P which, for any given MRM input ι, gives as a global optimum the output of the MRM on ι
   Make sure P has a unique global optimum
4. Does this prove that MP is a Turing-complete language?
5. Is P linear? If not, can you reformulate P *exactly* so it becomes linear?
6. Test your formulation P on the MRM isfactorof using AMPL and CPLEX (if P is linear) or BARON (otherwise)
7. Change P so it finds the input (a, b) yielding the fastest execution. What about the slowest execution?

Finding an odd perfect number

- A number is *perfect* if it is the sum of all its proper divisors (i.e. all aside from n itself)
  e.g. $6 = 1 \times 2 \times 3 = 1 + 2 + 3$; the next is 28
- Every perfect number found so far is even
- **Conjecture** α: there are no odd perfect numbers
- Let A be the set of all odd perfect numbers
  - is A recursively enumerable?
  - do you think A is decidable or undecidable?
  - do you think α has a proof in PA1?
- Exhibit a MP formulation which, if infeasible, proves that α is false. If your formulation has infinitely many variables or constraints, now provide a *finite* one