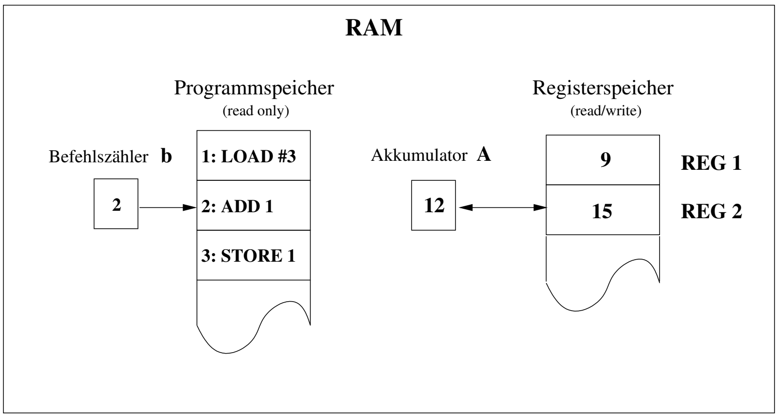
Technische Universität München
Institut für Informatik
Lehrstuhl für Effiziente Algorithmen
Dr. P. Ullrich

WS2001/02

Wissenswertes zur Registermaschine (RAM)

Aufbau einer RAM: (**RAM**=**R**andom **A**ccess **M**achine)



Die Registermaschine besteht aus

- abzählbar vielen Registern REG 1, REG 2, ... und einem ausgezeichneten Register, dem *Akkumulator* A. Jedes Register kann eine beliebig große ganze Zahl speichern.
- einem nur-lese Speicher, in dem das auszuführende Programm gespeichert ist. Beginnend mit 1 sind die einzelnen Programmzeilen (pro Zeile nur ein Befehl) fortlaufend durchnumeriert.
- einem *Befehlszähler* b, der auf den momentan zu bearbeitenden Befehl zeigt. Im Befehlszähler kann jede beliebig große natürliche Zahl stehen.

Die Registermaschine kann eine gewisse Anzahl von Maschinenbefehlen verarbeiten. Ein Machinenbefehl hat die allgemeine Form

Operator ␣ Operand (bzw. kein Operand)

Operatoren: LOAD, STORE, ADD, SUB, SHIFTL, SHIFTR, GOTO, IF $\square$ 0 GOTO, END
mit $\square \in \{=, \neq, <, \leq, \geq, >\}$.

Operanden:	$\#i$	Zahl $i \in \mathbb{Z}$	
	i	Register $i \in \mathbb{N}$	(direkte Adressierung)
	$*i$	Register von Register $i \in \mathbb{N}$	(indirekte Adressierung)

Schreibweisen:
Der Inhalt des Registers REG i (bzw. des Akkumulators A) wird mit $c(i)$ (bzw. $c(\mathrm{A})$) bezeichnet,

d.h. $c(i), c(\mathrm{A}) \in \mathbb{Z}$. Entsprechend bezeichnet $c(c(i))$ den Inhalt von Register REG $c(i)$.
Die Symbolik REG i $\leftarrow q$ (bzw. $\mathrm{A} \leftarrow q$) bedeutet, daß die ganze Zahl $q \in \mathbb{Z}$ in das Register REG i (bzw. in den Akkumulator A) gespeichert wird.

Erläuterung der Maschinenbefehle:

	$\#i$	i	$*i$	kein Operand
LOAD	$\mathrm{A} \leftarrow i$	$\mathrm{A} \leftarrow c(i)$	$\mathrm{A} \leftarrow c(c(i))$	–
STORE	–	REG $i \leftarrow c(\mathrm{A})$	REG $c(i) \leftarrow c(\mathrm{A})$	–
ADD	$\mathrm{A} \leftarrow c(\mathrm{A}) + i$	$\mathrm{A} \leftarrow c(\mathrm{A}) + c(i)$	$\mathrm{A} \leftarrow c(\mathrm{A}) + c(c(i))$	–
SUB	$\mathrm{A} \leftarrow c(\mathrm{A}) - i$	$\mathrm{A} \leftarrow c(\mathrm{A}) - c(i)$	$\mathrm{A} \leftarrow c(\mathrm{A}) - c(c(i))$	–
SHIFTL	–	–	–	$\mathrm{A} \leftarrow 2 * c(\mathrm{A})$
SHIFTR	–	–	–	$\mathrm{A} \leftarrow \lfloor c(\mathrm{A})/2 \rfloor$
GOTO	–	$\mathrm{b} \leftarrow i$	–	–
IF $\square$ 0 GOTO	– –	$\mathrm{b} \leftarrow i$, falls $c(\mathrm{A})\square 0$, b++, sonst	– –	– –
END	–	–	–	Berechnung stoppt

Funktionsweise einer RAM:

Zu Beginn der Berechnung enthält der Befehlszähler b den Wert 1. Die Eingabedaten $x_1, \ldots, x_m \in \mathbb{Z}$ sind in den Registern REG $1, \ldots,$ REG m gespeichert.
Nach Ausführung eines Befehls wird der Befehlszähler um Eins erhöht (b++) oder entsprechend einer GOTO-Anweisung auf einen bestimmten Wert gesetzt, bis END erreicht ist.
Die Ausgabedaten (=Resultate der Berechnung) $y_1, \ldots, y_n \in \mathbb{Z}$ befinden sich am Ende der Berechnung in den Registern REG $1, \ldots,$ REG n.

Logarithmische Kosten:

Zur abkürzenden Schreibweise definieren wir zunächst das logarithmische Kostenmaß cost(op) eines Operanden op:
(Erinnerung: Für $0 \neq i \in \mathbb{Z}$ ist $l(i) := \lfloor \log |i| \rfloor + 1$, und $l(0) = 1$)

$$\begin{aligned} \mathrm{cost}(\#i) &:= l(i) \\ \mathrm{cost}(i) &:= l(i) + l(c(i)) \\ \mathrm{cost}(*i) &:= l(i) + l(c(i)) + l(c(c(i))) \end{aligned}$$

Übersicht über die logarithmischen Kosten einer RAM:

Befehl	Logarithmische Kosten
LOAD op	cost(op)
STORE i	$l(c(\mathrm{A})) + l(i)$
STORE $*i$	$l(c(\mathrm{A})) + l(i) + l(c(i))$
ADD op	$l(c(\mathrm{A})) + \mathrm{cost(op)}$
SUB op	$l(c(\mathrm{A})) + \mathrm{cost(op)}$
SHIFTL	$l(c(\mathrm{A}))$
SHIFTR	$l(c(\mathrm{A}))$
GOTO	1
IF $\square$ 0 GOTO	$l(c(\mathrm{A}))$
END	1