One pebble versus log n bits

Viliam Geffert

P.J. Šafárik University, Košice, Slovakia

Carlo Mereghetti   *Giovanni Pighizzini*

Università degli Studi, Milano, Italy

A non-conventional Turing machine: the pebble machine

Pebble machine = Turing machine +

Chang, Ibarra, et al., 1986

Investigations on space bounded computation

- Space constructibility
- Space bounded recognition power

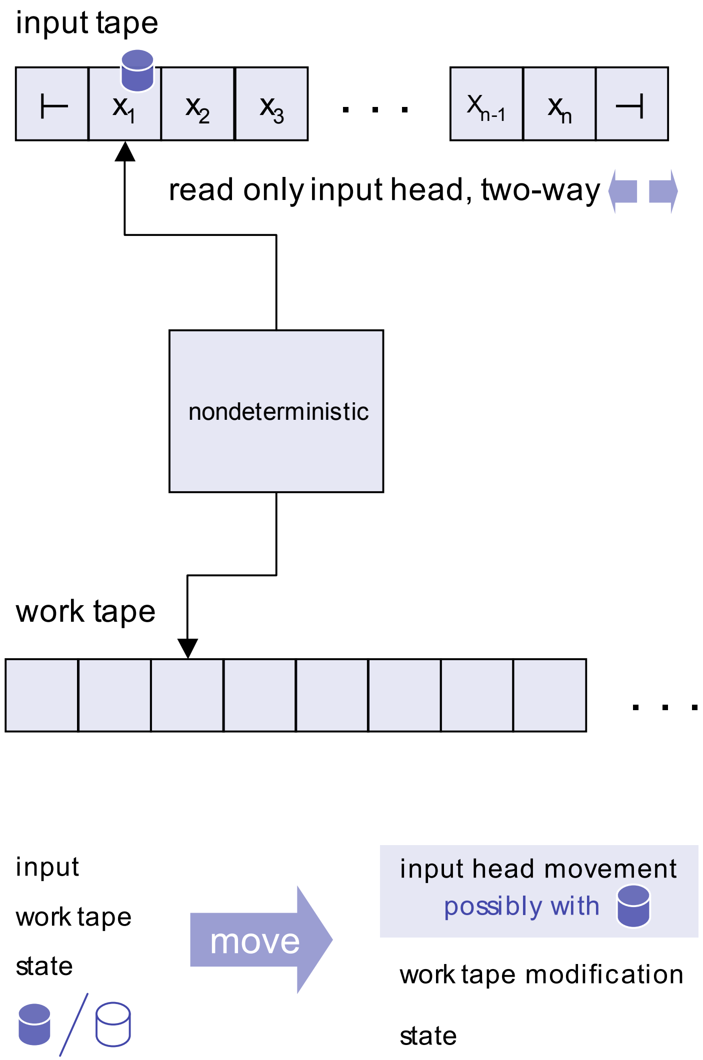
Sometimes is useful to save space, sometimes it is not

Our contribution on pebbling

When pebble does not help to save space:

- A language hard for pebbling
- Pebbling for nonregular languages

Pebble machine



Space notions

- Strong s(n) space: any computation on any input of length n uses at most s(n) work tape cells
- Weak s(n) space: for any accepted input of length n, there exists an accepting computation using at most s(n) work tape cells

strong = weak for s(n) fully space constructible

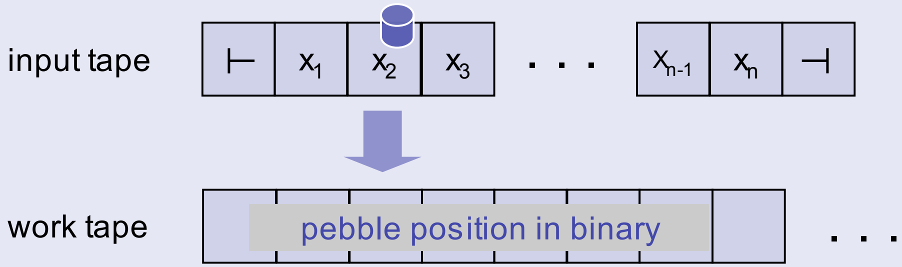
There exists a deterministic Turing machine that, on any input of length n, uses s(n) work tape cells

- All “normal” function above log n are fully space constructible
- No o(log n) unbounded non-decreasing function is fully space constructible

strong ≠ weak in the sublogarithmic space world

Space lower bounds for nonregular acceptance

Some facts on pebble machines



$s(n) \in \Omega(\log n)$ → 

$s(n) \in \Omega(\log\log n) \cap o(\log n)$ → where to appreciate the power of pebbling

$s(n) \in o(\log\log n)$ → 

Theorem
Pebble machines working in $o(\log\log n)$ space recognize regular languages only

useful to:

- fix input positions
- delimit input portions
- counting …

→ typical log-space consuming tasks

Can we go log-space by pebbling below log-space?

Language recognition: $a^n b^n$

- accepted in strong loglog n space on a (deterministic) pebble machine
- cannot be accepted by any Turing machine in weak o(log n) space

Space constructibility: loglog n is fully space constructible by pebble machines

Is this a general result? NO!

Two situation where pebbling does not reduce log-space:

→

- A language for which pebbling does not reduce the recognition space below log n
- Pebbling does not lower log n lower bound on space x input head reversals for nonregular acceptance

A language hard for pebbling

$$L = \left\{ x \in \{0,1\}^* : x = w0^*w^R \text{ and } |w| = \left\lfloor \sqrt{|x|} \right\rfloor \right\}$$

L can be accepted in (strong) log n space
by a 2-way deterministic Turing machine

L cannot be accepted in weak o(log n) space
by any pebble machine

$\Downarrow$

pebbling does not help in reducing the recognition space for L

A language hard for pebbling: $L = \left\{ x \in \{0,1\}^* : x = w0^*w^R \text{ and } |w| = \left\lfloor \sqrt{|x|} \right\rfloor \right\}$

L can be accepted in (strong) log n space by a 2-way deterministic Turing machine

```
input( x )                // with | x | = n

p := floor(sqrt( n ))

i := 1
while i <= p do           // check w = w^R
   begin
      if x_i <> x_{n-i+1} then
         reject
      i := i +1
   end

i := i +1
while i <= n - p do       // check the existence of
   begin                  //  n - 2√n zeroes between
      if x_i <> '0' then  // w and w^R
         reject
      i := i +1
   end

accept
```

Space requirements

- Compute and write in binary **floor(sqrt(** n **))**
- Manage counters fixing positions

log n

one work tape cells

Improvements

- Storing **floor(sqrt(** n **))** in log n / 2 cells
- Smart counters managing in log n / 2 cells

A language hard for pebbling: $L = \left\{ x \in \{0,1\}^* : x = w0^*w^R \text{ and } |w| = \left\lfloor \sqrt{|x|} \right\rfloor \right\}$

L cannot be accepted in weak o(log n) space by any pebble machine

By contradiction: There exists a pebble machine M accepting L in weak $s(n) \in o(\log n)$ space

We will fool M

Memory state of M: (i , q , w)

- w → work tape content
- q → state
- i → work tape head position

$H^{(n)}$ = { memory states of M using no more than s(n) work tape cells }

- $|H^{(n)}| \le d^{s(n)} = \mu(n)$
- $s(n) \in o(\log n) \Longrightarrow \mu^2(n) \in o(\sqrt{n})$

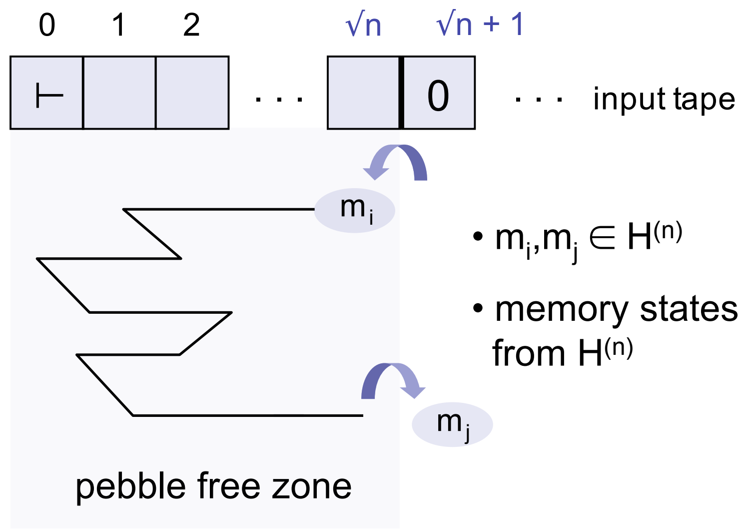
$L^{(n)} = \{ x \in L : |x| = n \}$

- $|L^{(n)}| = 2^{\lfloor \sqrt{n} \rfloor}$
- weak space $\Longrightarrow$ any string in $L^{(n)}$ is accepted by an accepting computation whose memory states are from $H^{(n)}$

Entering and exiting input strings prefixes and suffixes

$x = w0^*w^R \in L^{(n)}$

- the $\mu(n) \times \mu(n)$ boolean matrix $P^{(n)}$ whose (i,j)th entry is 1 if and only if:
- the $\mu(n) \times \mu(n)$ boolean matrix $S^{(n)}$ giving the symmetrical infos on w^R



- $m_i, m_j \in H^{(n)}$
- memory states from $H^{(n)}$

Some counting facts:

- There exist at most $2^{\mu^2(n)}$ different $P^{(n)}$ and $2^{\mu^2(n)}$ different $S^{(n)}$ matrices

- $|H^{(n)}| \leq d^{s(n)} = \mu(n)$
- $s(n) \in o(\log n) \Rightarrow \mu^2(n) \in o(\sqrt{n})$
- $|L^{(n)}| = 2^{\lfloor\sqrt{n}\rfloor}$

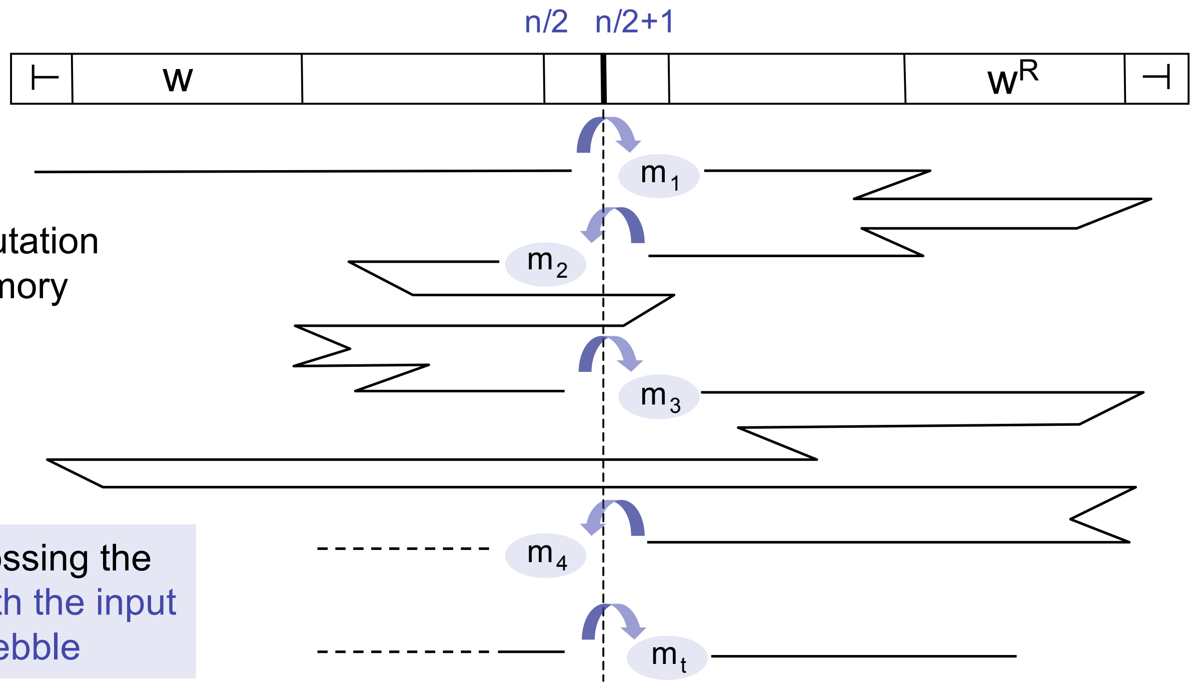
pigeonhole

There is an arbitrarily large set $W^{(n)} \subseteq L^{(n)}$ of strings having the same $P^{(n)}$ and $S^{(n)}$ matrices

$$|W^{(n)}| \geq \frac{|L^{(n)}|}{2^{2\mu^2(n)}} = 2^{\lfloor\sqrt{n}\rfloor - 2\mu^2(n)}$$

Playing around the middle of input strings

$x = w0^*w^R \in W^{(n)}$



input tape

An accepting computation of M on x using memory states from $H^{(n)}$

m_i is reached by crossing the middle boundary with the input head carrying the pebble

$\gamma = m_1, m_2, \ldots, m_t$ → middle crossing sequence

Some counting facts:

- $t \leq 2\mu(n)$, otherwise γ can be shortened and still represent a middle crossing sequence of an accepting computation of M on x
- There exist at most $\mu(n)^{2\mu(n)+1}$ different middle crossing sequences
- $\mu^2(n) \in o(\sqrt{n})$

pigeonhole →

There exists an arbitrarily large set $V^{(n)} \subseteq W^{(n)} \subseteq L^{(n)}$ of strings having the same middle crossing sequence

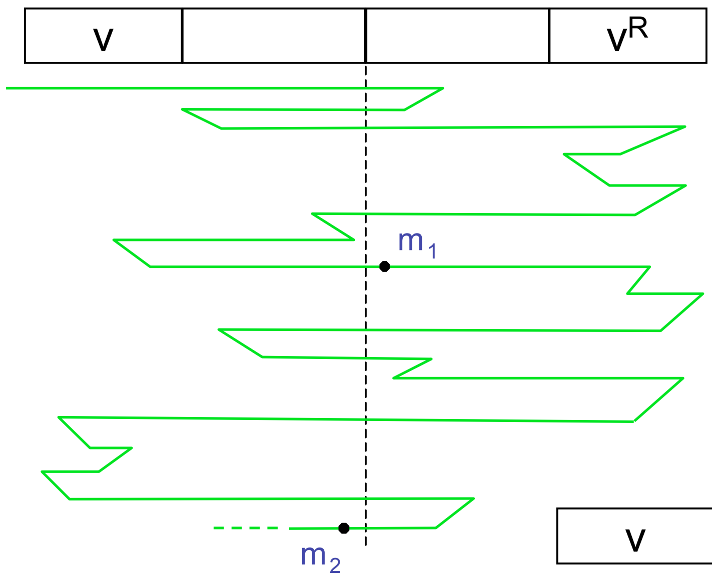
$$|V^{(n)}| \geq \frac{|W^{(n)}|}{\mu(n)^{2\mu(n)+1}} \geq 2^{\lfloor\sqrt{n}\rfloor - 3\mu^2(n)}$$

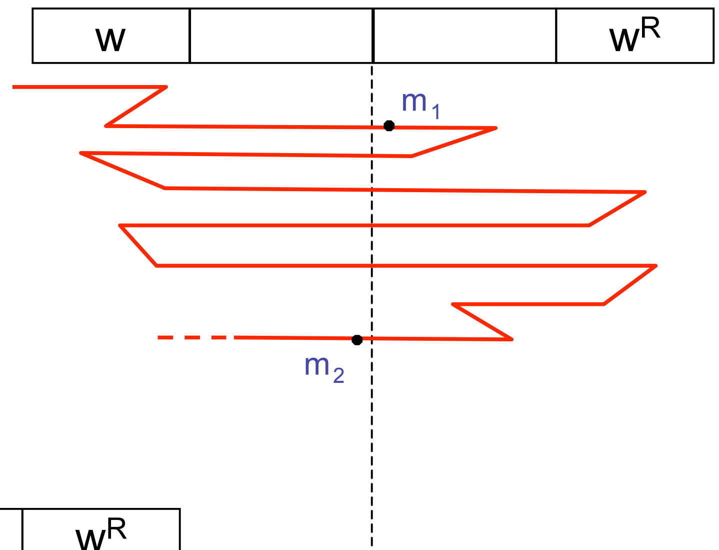
Fooling M: the contradiction

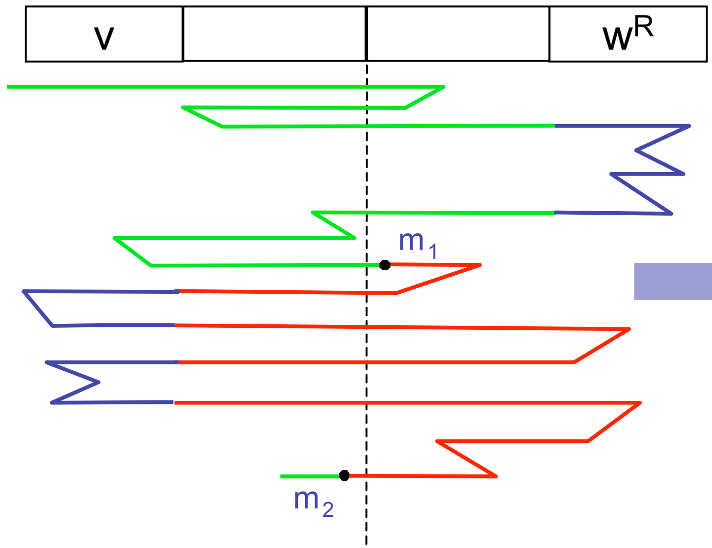
$v0^*v^R \in V^{(n)}$

$w0^*w^R \in V^{(n)}$

with $v \neq w$, sharing the same $P^{(n)}$ and $S^{(n)}$ matrices, and with accepting computations having memory states in $H^{(n)}$ and the same middle crossing sequence $m_1, m_2, \ldots$







an accepting computation for the string $v0^*w^R \notin L^{(n)}$

contradiction!

Pebbling on nonregular languages

Problem: What is the minimal amount of space x input head reversals for a pebble machine recognizing a nonregular language?

$s(n) \cdot i(n) \notin o(\ ?\ )$

Pebble machine working in strong s(n) space and i(n) input head reversals:
any computation on any input of length n uses at most s(n) work tape cells and i(n) input head reversals

For Turing machines:

- $s(n) \cdot i(n) \notin o(\log n)$
- Unary optimality: a unary language accepted within:
  - $s(n) \in O(\log\log n)$, the smallest possible
  - $i(n) \in O(\log n / \log\log n)$

Can pebbling reduce the logarithmic lower bound on $s(n) \cdot i(n)$ for nonregular acceptance?

NO! ⇒ Even for pebbling, $s(n) \cdot i(n) \notin o(\log n)$

Logarithmic lower bound on $s(n)\cdot i(n)$ for nonregular pebbling

M is a pebble machine working in strong $s(n)$ and $i(n)$

pebble crossing sequence

$\Downarrow$

construct an equivalent nondeterministic TM M'
one-way and working in accept $O(s(n)\cdot i(n))$ space

$\Downarrow$

apply to M' the logarithmic space lower bound for
nondeterministic one-way accept TM

$\Downarrow$

$$s(n)\cdot i(n) \notin o(\log n)$$

Unary optimality: directly comes from that for Turing machines

An open problem: can pebbling be crucial in witnessing unary optimality?

Is there a unary nonregular language accepted by a strong pebble machine such that:

- $s(n) \in O(\log\log n)$, the smallest possible for pebbling
- $i(n) \in O(\log n / \log\log n)$
- $s(n) \in \Omega(\log\log n)$ without pebble, i.e., pebble is crucial in space saving?

Thank you for your attention!