space complexity

amir yehudayoff

computational resources

time

memory (space)

data

energy

randomness

....

computational complexity

devices have costs

problems have complexities (minimum cost for solving)

space: examples

$\sqrt{2}, \pi$

memory of programs

space complexity of Palindromes

tape of a Turing machine

definition: cost

a TM M and input $x \in \{0,1\}^*$

SPACE(M, x) is the number of locations on the (working) tape M visits during the run on x (could be ∞)

for $n \in \mathbb{N}$

$$\text{SPACE}(M, n) = \max_{x \in \{0,1\}^n} \text{SPACE}(M, x)$$

remarks

space is at most time

memory locations can be used more than once “for free”

do not count access to input (sublinear space)

classes

let $S : \mathbb{N} \to \mathbb{N}$

the language $L \subseteq \{0,1\}^*$ is in SPACE($S(n)$) if there is a TM M that decides L so that

$$\text{SPACE}(M, n) = O(S(n))$$

e.g. SPACE(n)

space constructible

from now on we assume that there is a TM M that given 1^n as input computes $S(n)$ using space $O(S(n))$

reasonable functions are such but not all functions

non determinism

a powerful resource: non determinism, “guess”, $\exists$

the space cost of a NTM M on x is the maximum space in the run of M on x over all (non-det.) choices

NSPACE($S(n)$)

space and time

theorem

for every $S : \mathbb{N} \to \mathbb{N}$ so that $S(n) \geq \log_2 n$

$$\begin{aligned} \text{DTIME}(S(n)) &\subseteq \text{SPACE}(S(n)) \\ &\subseteq \text{NSPACE}(S(n)) \subseteq \text{DTIME}(2^{O(S(n))}) \end{aligned}$$

first two inclusions ...

third ...

configuration graphs

directed graph $G_{M,x}$ for TM M and input x

vertices are the "configurations" (i.e., a vertex v encodes a possible state of the computation but not x)

there is an edge (v, u) if M moves from state v to state u in one time step

there is a single "accept vertex" (w.l.o.g.)

remarks

vertices do not depend on x but edges do (they must)

if M is deterministic, all out-degrees are one

if M is non-deterministic, all out-degrees are at most two

loops correspond to computations that do not terminate

observations

M accepts $x \in \{0,1\}^n$ iff there is a path from the initial state to the accept vertex in $G_{M,x}$

described vertices by $B := O(\text{SPACE}(M,x) + \log n)$ bits

the number of vertices is at most 2^B

the edges

for every TM or NTM M and $x \in \{0,1\}^n$, there is a CNF formula $\varphi = \varphi_{M,x}$ so that for every two strings v, u

$\varphi(v,u) = 1$ iff v, u encode vertices and (v,u) is edge in $G_{M,x}$

the size of φ is $\leq O(\text{SPACE}(M,x) + \log n)$

reason AND of many local checks (Cook-Levin)

simulating NTIME

think of a computation as walk on graph

given $M \in \text{NSPACE}(S(n))$, for input $x \in \{0,1\}^n$ need to decide if there is a path from initial state to accepting state in $G_{M,x}$

BFS in time $poly(|G_{M,x}|) \leq 2^{O(S(n)+\log n)}$

classification

computational complexity classifies problems according to resources

completeness

class C of problems

X is complete for C if $X \in C$ and $Y \leq X$ for all $Y \in C$

poly space

$$\text{PSPACE} = \bigcup_{k \in \mathbb{N}} \text{SPACE}(n^k)$$

$$\text{NPSPACE} = \bigcup_{k \in \mathbb{N}} \text{NSPACE}(n^k)$$

relationships

$$\mathrm{P} \subseteq \mathrm{NP} \subseteq \mathrm{PSPACE}$$

check all assignments for a formula in polynomial space (and exponential time)

do not know if $\mathrm{P} = \mathrm{PSPACE}$ but if $\mathrm{P} = \mathrm{PSPACE}$ then $\mathrm{P} = \mathrm{NP}$

PSPACE **completeness**

$C \subseteq \{0,1\}^*$ is PSPACE-complete if $C \in$ PSPACE and $L \leq_p C$ for every $L \in$ PSPACE

$X \leq_p Y$ if there is $f : \{0,1\}^* \to \{0,1\}^*$ in poly-time so that $x \in X$ iff $f(x) \in Y$

canonical completeness

$$\{\langle M, x, 1^s \rangle : M(x) = 1, \text{SPACE}(M, x) \leq s\}$$

but we want “natural” classes

boolean formulas

variables $x_1, \ldots, x_n$

the expressions $x_1, \ldots, x_n$ and $0, 1$ are formulas

if F is a formula then $(\neg F)$ is a formula

if F_1, F_2 are formulas then $(F_1 \wedge F_2)$ and $(F_1 \vee F_2)$ are formulas

remarks

formulas are rooted labelled binary trees

formulas compute boolean functions $\{0,1\}^n \to \{0,1\}$
  every boolean function has a formula
  formula complexity

if F has size s then can be described by $O(s)$ bits

totally quantified boolean formula

a TQBF is an expression of the form

$$E = Q_1 x_1 Q_2 x_2 \ldots Q_n x_n \ \varphi$$

where φ is formula over $x_1, \ldots, x_n$ and $Q_i \in \{\forall, \exists\}$ for all i

remarks

$$Q_1 x_1 Q_2 x_2 \ldots Q_n x_n \; \varphi$$

all variables are quantified

each x_i takes values in $\{0, 1\}$

a TQBF has a truth value in $\{0, 1\}$

game theory $\exists w_1 \forall b_1 \exists w_2 \forall b_2 \ldots$

TQBF

theorem

$$\mathit{TQBF} = \{\langle E \rangle : E \text{ is a true TQBF}\}$$

is PSPACE-complete

TQBF $\in$ PSPACE

the algorithm A is recursive—reuses space

(base) no quantifiers
(step) input $\forall x \psi(x)$ for formula ψ

recursively set $y_0 = A(\psi(0))$
write y_0 and delete the rest of working memory
set $y_1 = A(\psi(1))$
output $y_0 \wedge y_1$

the case of $\exists x \psi(x)$ is similar

TQBF $\in$ PSPACE

the algorithm A is recursive

(step) input $\forall x \psi(x)$

recursively set $y_0 = A(\psi(0))$
write y_0 and delete the working memory
set $y_1 = A(\psi(1))$
output $y_0 \wedge y_1$

memory size is
O(the number of quantifiers plus the size of the inner formula)

TQBF is PSPACE-hard

$L \in$ PSPACE and M a poly-space TM for L and input x

goal: construct a TQBF $\psi = \psi_x$ so that

$$M(x) = 1 \iff \psi = 1$$

TQBF is PSPACE-hard

TQBF ψ so that $M(x) = 1 \iff \psi = 1$

recall $\varphi_{M,x}$ that checks if v, u is an edge in $G = G_{M,x}$

TQBF is PSPACE-hard

TQBF ψ so that $M(x) = 1 \iff \psi = 1$

recall $\varphi_{M,x}$ that checks if v, u is an edge in $G = G_{M,x}$

inductively define $\phi_0 = \varphi_{M,x}$ and

$$\phi_i(v,u) = \exists w \, \forall a, b \quad ((a = v) \wedge (b = w)) \vee ((a = w) \wedge (b = u)) \rightarrow \phi_{i-1}(a,b)$$

TQBF is PSPACE-hard

TQBF ψ so that $M(x) = 1 \iff \psi = 1$

recall $\varphi_{M,x}$ that checks if v, u is an edge in $G = G_{M,x}$

inductively define $\phi_0 = \varphi_{M,x}$ and

$$\phi_i(v,u) = \exists w\ \forall a,b \quad ((a = v) \wedge (b = w)) \vee ((a = w) \wedge (b = u)) \to \phi_{i-1}(a,b)$$

claim checks $dist_G(v,u) \leq 2^i$ and has size $O(i \cdot \log |G|)$

TQBF is PSPACE-hard

TQBF ψ so that $M(x) = 1 \iff \psi = 1$

recall $\varphi_{M,x}$ that checks if v, u is an edge in $G = G_{M,x}$

inductively define $\phi_0 = \varphi_{M,x}$ and

$$\phi_i(v, u) = \exists w \, \forall a, b \quad ((a = v) \wedge (b = w)) \vee ((a = w) \wedge (b = u)) \rightarrow \phi_{i-1}(a, b)$$

claim checks $dist_G(v, u) \leq 2^i$ and has size $O(i \cdot \log |G|)$

set

$$\psi = \phi_{O(\log |G|)}(v_0, v_{accept})$$

power of TQBF

$$\exists w \, \forall a, b$$
$$((a = v) \wedge (b = w)) \vee ((a = w) \wedge (b = u)) \rightarrow \phi_{i-1}(a, b)$$

versus

$$\exists w$$
$$\phi_{i-1}(u, w) \wedge \phi_{i-1}(w, v)$$

summary

$$\mathit{TQBF} = \{\langle E \rangle : E \text{ is a true TQBF}\}$$

is PSPACE-complete

specifically, deciding the truth value of a TQBF is as hard as any poly-memory problem

observation

we never used out-degrees are one

so it applies to non-deterministic computations

corollary

PSPACE $=$ NPSPACE

non-deterministic space

theorem **[Savitch 1970]**

if $S(n) \geq \log_2 n$ then

$$\text{NSPACE}(S(n)) \subseteq \text{SPACE}((S(n))^2)$$

similar proof—TQBF of size $\approx \log^2 |G|$

log space

allowing space s allows to check $\exp(s)$ options

the space analogs of P and NP are

$$\mathsf{L} = \mathsf{SPACE}(\log n)$$

$$\mathsf{NL} = \mathsf{NSPACE}(\log n)$$

reductions

correct notion of reductions for log space?

$f : \{0,1\}^* \to \{0,1\}^*$ is implicitly log-space computable (ILC) if

i. there is $C > 0$ so that $|f(x)| \leq C|x|^C$ for all x

ii. $\{\langle x, i \rangle : f(x)_i = 1\} \in L$

iii. $\{\langle x, i \rangle : i \leq |f(x)|\} \in L$

log space reductions

$A \leq_\ell B$ if there is ILC f so that $x \in A$ iff $f(x) \in B$ for all x

properties (exercise)

if $A \leq_\ell B$ and $B \in \mathsf{L}$ then $A \in \mathsf{L}$

if $A \leq_\ell B$ and $B \leq_\ell C$ then $A \leq_\ell C$

intuitive but not obvious—space must be reused

NL

$C \subseteq \{0,1\}^*$ is NL-complete if $C \in NL$ and $L \leq_\ell C$ for every $L \in$ NL

theorem

the language *PATH*

$\{\langle G, s, t\rangle : G$ digraph, $s, t \in V(G)$, there is a path from s to $t\}$

is *NL*-complete

NL

theorem

PATH is *NL*-complete

sketch of proof...

$PATH \in NL$

w is a path from s of length $\leq |G|$

accept only if w reaches t

when walking along w we forget where we came from

PATH is *NL*-hard

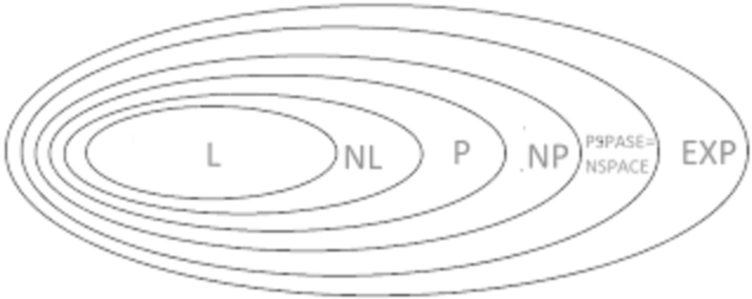
if $L \in NL$ then there is log-space NTM N for it

the digraph $G_{N,x}$

the reduction

$$f(x) = \langle G_{N,x}, v_0, v_{\text{accept}} \rangle$$

summary



coNL

$$coNL = \{L : L^c \in NL\}$$

remark

deterministic classes are closed for “co” but not obvious (or false) for non-deterministic

co for log-space

theorem **[Immerman-Szelepcsenyi]**

$$NL = coNL$$

exercises

suffices to prove $coNL \subseteq NL$

suffices to prove $PATH^c \in NL$

$PATH^c \in NL$

need a witness that "proves" there is no path

a log space M so that for every (G, s, t)

if $\langle G, s, t\rangle \notin PATH$ then $\exists w : M(\langle G, s, t\rangle, w) = 1$

if $\langle G, s, t\rangle \in PATH$ then $\forall w : M(\langle G, s, t\rangle, w) = 0$

remark $|w| \leq poly$

$PATH^c \in NL$ part I

a log space M so that for every (G, s, t)

if c is number of vertices reachable from s then

if $\langle G, s, t\rangle \notin PATH$ then $\exists w : M(\langle G, s, t, c\rangle, w) = 1$

if $\langle G, s, t\rangle \in PATH$ then $\forall w : M(\langle G, s, t, c\rangle, w) = 0$

$PATH^c \in NL$ part I

given G, s, t and $c = |reach(s)|$

for each $v \in V$

$$b_v = 1[v \in reach(s)]$$
$$p_v = \begin{cases} \text{path } s \to v & b_v = 1 \\ \bot & b_v = 0 \end{cases}$$

set $w = ((b_v, p_v) : v \in V)$

M checks in log space[1] (verify...)

1. $\sum_v b_v = c$
2. $b_t = 0$
3. for all v if $b_v = 1$ then p_v is valid

---

[1] input w is not counted as memory

$PATH^c \in NL$ part II

given G, s, t and $c = |reach(s)|$

remains to certify c

let $V_i = \{v : dist(s, v) \leq i\}$

so $|V_0| = 1$ and $|V_n| = c$

$PATH^c \in NL$ part II

given G, s, t and $c = |reach(s)|$

remains to certify c

let $V_i = \{v : dist(s, v) \leq i\}$

so $|V_0| = 1$ and $|V_n| = c$

certify $|V_{i+1}|$ **given** $|V_i|$

$PATH^c \in NL$ part II

$V_i = \{v : dist(s, v) \leq i\}$

goal certify $|V_{i+1}| = c_{i+1}$ given $|V_i| = c_i$

PATH$^c \in NL$ **part II**

$V_i = \{v : dist(s, v) \leq i\}$

goal certify $|V_{i+1}| = c_{i+1}$ given $|V_i| = c_i$

$v \in V_{i+1}$: path p_v of length $\leq i + 1$

$PATH^c \in NL$ part II

$V_i = \{v : dist(s, v) \leq i\}$

goal certify $|V_{i+1}| = c_{i+1}$ given $|V_i| = c_i$

$v \in V_{i+1}$: path p_v of length $\leq i + 1$

$v \notin V_{i+1}$: for all $u \in V_i$ there is no edge (u, v)

$PATH^c \in NL$ part II

$V_i = \{v : dist(s, v) \leq i\}$

goal certify $|V_{i+1}| = c_{i+1}$ given $|V_i| = c_i$

$v \in V_{i+1}$: path p_v of length $\leq i + 1$

$v \notin V_{i+1}$: for all $u \in V_i$ there is no edge (u, v)

certify $1[v \in V_{i+1}]$ for all v and check size

summary $PATH^c \in NL$

two things we can do

count

verify that a path is correct

total size of w is $poly(|G|)$

log-space

theorem **[Immerman-Szelepcsenyi]**

$$NL = coNL$$

summary

space

poly-space TQBF

log-space PATH

computation as a huge graph with simple edges