Complements of Nondeterministic Classes

- From p. 133, we know R, RE, and coRE are distinct.
  - coRE contains the complements of languages in RE, *not* the languages not in RE.
- Recall that the **complement** of L, denoted by $\bar{L}$, is the language $\Sigma^* - L$.
  - SAT COMPLEMENT is the set of unsatisfiable boolean expressions.
  - HAMILTONIAN PATH COMPLEMENT is the set of graphs without a Hamiltonian path.

The Co-Classes

- For any complexity class $\mathcal{C}$, co$\mathcal{C}$ denotes the class

$$\{L : \bar{L} \in \mathcal{C}\}.$$

- Clearly, if $\mathcal{C}$ is a *deterministic* time or space *complexity class*, then $\mathcal{C} = \text{co}\mathcal{C}$.
  - They are said to be **closed under complement**.
  - A deterministic TM deciding L can be converted to one that decides $\bar{L}$ within the same time or space bound by reversing the "yes" and "no" states.
- Whether nondeterministic classes for time are closed under complement is not known (p. 79).

Comments

- As
$$\text{co}\mathcal{C} = \{L : \bar{L} \in \mathcal{C}\},$$
$L \in \mathcal{C}$ if and only if $\bar{L} \in \text{co}\mathcal{C}$.
- But it is *not* true that $L \in \mathcal{C}$ if and only if $L \notin \text{co}\mathcal{C}$.
  - co$\mathcal{C}$ is not defined as $\bar{\mathcal{C}}$.
- For example, suppose $\mathcal{C} = \{\{2, 4, 6, 8, 10, \ldots\}\}$.
- Then co$\mathcal{C} = \{\{1, 3, 5, 7, 9, \ldots\}\}$.
- But $\bar{\mathcal{C}} = 2^{\{1,2,3,\ldots\}^*} - \{\{2, 4, 6, 8, 10, \ldots\}\}$.

The Quantified Halting Problem

- Let $f(n) \geq n$ be proper.
- Define

$$H_f = \{M; x : M \text{ accepts input } x \text{ after at most } f(|\,x\,|) \text{ steps}\},$$

  where M is deterministic.
- Assume the input is binary.

$H_f \in \mathsf{TIME}(f(n)^3)$

- For each input $M; x$, we simulate M on x with an alarm clock of length $f(|\, x\,|)$.
  - Use the single-string simulator (p. 57), the universal TM (p. 118), and the linear speedup theorem (p. 64).
  - Our simulator accepts $M; x$ if and only if M accepts x before the alarm clock runs out.
- From p. 63, the total running time is $O(\ell_M k_M^2 f(n)^2)$, where ℓ_M is the length to encode each symbol or state of M and k_M is M's number of strings.
- As $\ell_M k_M^2 = O(n)$, the running time is $O(f(n)^3)$, where the constant is independent of M.

$H_f \notin \mathsf{TIME}(f(\lfloor n/2 \rfloor))$

- Suppose TM M_{H_f} decides H_f in time $f(\lfloor n/2 \rfloor)$.
- Consider machine $D_f(M)$:

$$\textbf{if } M_{H_f}(M;M) = \text{"yes" } \textbf{then } \text{"no" } \textbf{else } \text{"yes"}$$

- D_f on input M runs in the same time as M_{H_f} on input $M;M$, i.e., in time $f(\lfloor \frac{2n+1}{2} \rfloor) = f(n)$, where $n = |\, M \,|$.[a]

[a] A student pointed out on October 6, 2004, that this estimation omits the time to write down $M;M$.

The Proof (concluded)

- First,

$$
\begin{aligned}
& D_f(D_f) = \text{"yes"} \\
\Rightarrow \quad & D_f; D_f \not\in H_f \\
\Rightarrow \quad & D_f \text{ does not accept } D_f \text{ within time } f(\mid D_f \mid) \\
\Rightarrow \quad & D_f(D_f) \neq \text{"yes"} \\
\Rightarrow \quad & D_f(D_f) = \text{"no"}
\end{aligned}
$$

a contradiction

- Similarly, $D_f(D_f) =$ "no" $\Rightarrow D_f(D_f) =$ "yes."

The Time Hierarchy Theorem

Theorem 16 *If $f(n) \geq n$ is proper, then*

$$\text{TIME}(f(n)) \subsetneq \text{TIME}(f(2n+1)^3).$$

- The quantified halting problem makes it so.

Corollary 17 $\text{P} \subsetneq \text{EXP}$.

- $\text{P} \subseteq \text{TIME}(2^n)$ because $\text{poly}(n) \leq 2^n$ for n large enough.
- But by Theorem 16,

$$\text{TIME}(2^n) \subsetneq \text{TIME}((2^{2n+1})^3) \subseteq \text{TIME}(2^{n^2}) \subseteq \text{EXP}.$$

- So $\text{P} \subsetneq \text{EXP}$.

The Space Hierarchy Theorem

Theorem 18 (Hennie and Stearns (1966)) *If $f(n)$ is proper, then*

$$\mathrm{SPACE}(f(n)) \subsetneq \mathrm{SPACE}(f(n) \log f(n)).$$

Corollary 19 $\mathrm{L} \subsetneq \mathrm{PSPACE}$.

Nondeterministic Time Hierarchy Theorems

Theorem 20 (Cook (1973)) *If $f(n)$ is proper, then*

$$\text{NTIME}(n^r) \subsetneq \text{NTIME}(n^s)$$

whenever $1 \leq r < s$.

Theorem 21 (Seiferas, Fischer, and Meyer (1978)) *If $T_1(n), T_2(n)$ are proper, then*

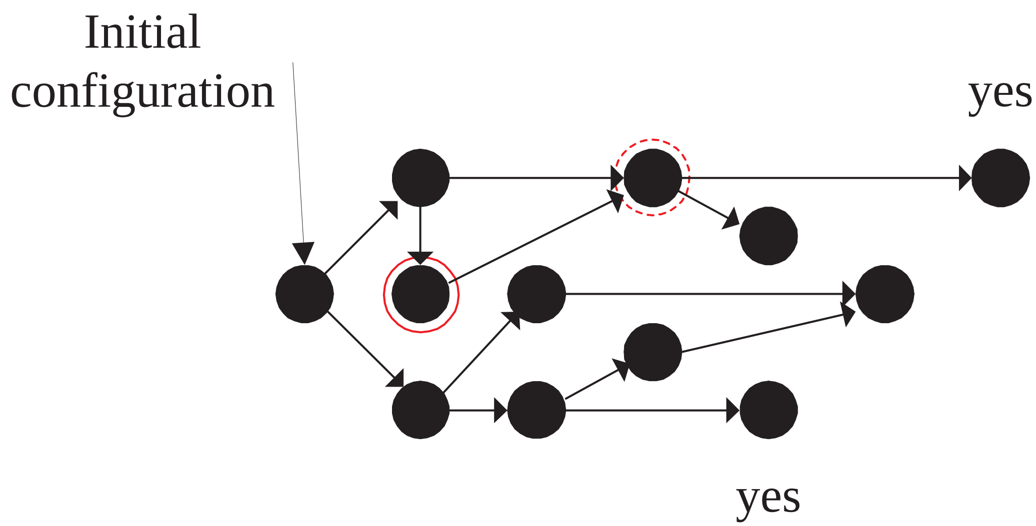
$$\text{NTIME}(T_1(n)) \subsetneq \text{NTIME}(T_2(n))$$

whenever $T_1(n+1) = o(T_2(n))$.

The Reachability Method

- The computation of a time-bounded TM can be represented by a directed graph.
- The TM's configurations constitute the nodes.
- Two nodes are connected by a directed edge if one yields the other.
- The start node representing the initial configuration has zero in degree.
- When the TM is nondeterministic, a node may have an out degree greater than one.

Illustration of the Reachability Method



Relations between Complexity Classes

Theorem 22 *Suppose $f(n)$ is proper. Then*

1. SPACE$(f(n)) \subseteq$ NSPACE$(f(n))$,
   TIME$(f(n)) \subseteq$ NTIME$(f(n))$.
2. NTIME$(f(n)) \subseteq$ SPACE$(f(n))$.
3. NSPACE$(f(n)) \subseteq$ TIME$(k^{\log n+f(n)})$.

- Proof of 2:
  - Explore the computation *tree* of the NTM for "yes."
  - Specifically, generate a $f(n)$-bit sequence denoting the nondeterministic choices over $f(n)$ steps.

Proof of Theorem 22(2)

- (continued)
  - Simulate the NTM based on the choices.
  - Recycle the space and then repeat the above steps until a "yes" is encountered or the tree is exhausted.
  - Each path simulation consumes at most $O(f(n))$ space because it takes $O(f(n))$ time.
  - The total space is $O(f(n))$ because space is recycled.

Proof of Theorem 22(3)

- Let k-string NTM

$$M = (K, \Sigma, \Delta, s)$$

with input and output decide $L \in \text{NSPACE}(f(n))$.

- Use the reachability method on the configuration graph of M on input x of length n.
- A configuration is a $(2k+1)$-tuple

$$(q, w_1, u_1, w_2, u_2, \ldots, w_k, u_k).$$

Proof of Theorem 22(3) (continued)

- We only care about

$$(q, i, w_2, u_2, \ldots, w_{k-1}, u_{k-1}),$$

  where i is an integer between 0 and n for the position of the first cursor.

- The number of configurations is therefore at most

$$|K| \times (n+1) \times |\Sigma|^{(2k-4) f(n)} = O(c_1^{\log n + f(n)}) \qquad (1)$$

  for some c_1, which depends on M.

- Add edges to the configuration graph based on M's transition function.

Proof of Theorem 22(3) (concluded)

- $x \in L \Leftrightarrow$ there is a path in the configuration graph from the initial configuration to a configuration of the form ("yes", $i, \ldots$) [there may be many of them].
- This is REACHABILITY on a graph with $O(c_1^{\log n + f(n)})$ nodes.
- It is in TIME$(c^{\log n + f(n)})$ for some c because REACHABILITY $\in$ TIME(n^j) for some j and

$$\left[c_1^{\log n + f(n)} \right]^j = (c_1^j)^{\log n + f(n)}.$$

Space-Bounded Computation and Proper Functions

- In the definition of *space-bounded* computations earlier, the TMs are not required to halt at all.
- When the space is bounded by a proper function f, computations can be assumed to halt:
  - Run the TM associated with f to produce an output of length $f(n)$ first.
  - The space-bounded computation must repeat a configuration if it runs for more than $c^{\log n + f(n)}$ steps for some c (p. 196).
  - So we can prevent infinite loops during simulation by pruning any path longer than $c^{\log n + f(n)}$.

The Grand Chain of Inclusions

$$L \subseteq NL \subseteq P \subseteq NP \subseteq PSPACE \subseteq EXP.$$

- By Corollary 19 (p. 189), we know $L \subsetneq PSPACE$.
- The chain must break somewhere between L and PSPACE.[a]
- It is suspected that all four inclusions are proper.
- But there are no proofs yet.[b]

[a] Bill Gates (1996), "I keep bumping into that silly quotation attributed to me that says 640K of memory is enough."

[b] Carl Friedrich Gauss (1777–1855), "I could easily lay down a multitude of such propositions, which one could neither prove nor dispose of."

Nondeterministic Space and Deterministic Space

- By Theorem 4 (p. 84),

$$\text{NTIME}(f(n)) \subseteq \text{TIME}(c^{f(n)}),$$

  an exponential gap.

- There is no proof yet that the exponential gap is inherent.
- How about NSPACE vs. SPACE?
- Surprisingly, the relation is only quadratic—a polynomial—by Savitch's theorem.

Savitch's Theorem

Theorem 23 (Savitch (1970))

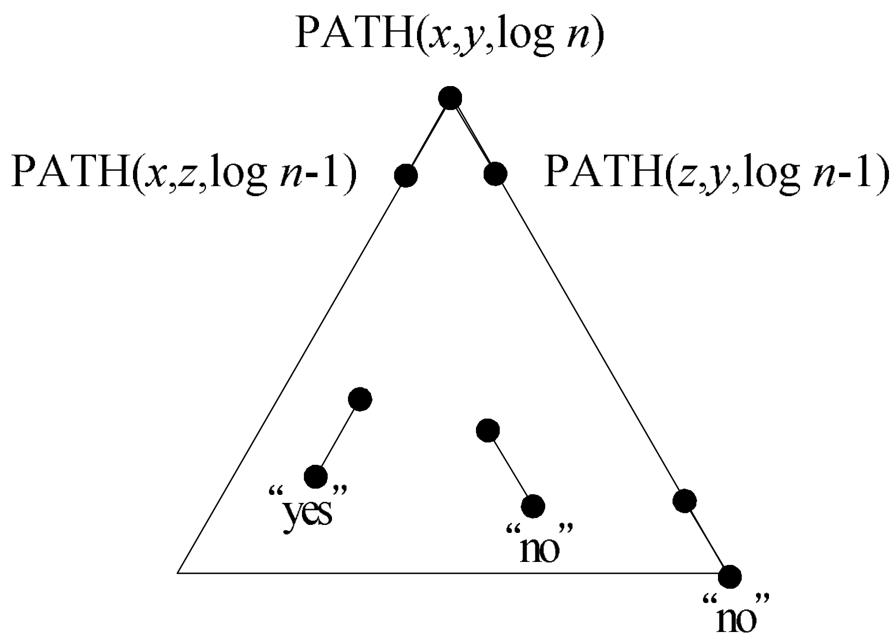
$$\textsc{reachability} \in \text{SPACE}(\log^2 n).$$

- Let $G(V, E)$ be a graph with n nodes.
- For $i \geq 0$, let
$$\text{PATH}(x, y, i)$$
mean there is a path from node x to node y of length at most 2^i.
- There is a path from x to y if and only if
$$\text{PATH}(x, y, \lceil \log n \rceil)$$
holds.

The Proof (continued)

- For $i > 0$, PATH(x, y, i) if and only if there exists a z such that PATH$(x, z, i-1)$ and PATH$(z, y, i-1)$.
- For PATH$(x, y, 0)$, check the input graph or if $x = y$.
- Compute PATH$(x, y, \lceil \log n \rceil)$ with a depth-first search on a graph with nodes (x, y, z, i)s (see next page).[a]
- Like stacks in recursive calls, we keep only the current path of (x, y, i)s.
- The space requirement is proportional to the depth of the tree: $\lceil \log n \rceil$.

[a]Contributed by Mr. Chuan-Yao Tan on October 11, 2011.



- Depth is $\lceil \log n \rceil$, and each node (x, y, z, i) needs space $O(\log n)$.
- The total space is $O(\log^2 n)$.

The Proof (concluded): Algorithm for PATH(x, y, i)

```
if i = 0 then
    if x = y or (x, y) ∈ E then
        return true;
    else
        return false;
    end if
else
    for z = 1, 2, ..., n do
        if PATH(x, z, i − 1) and PATH(z, y, i − 1) then
            return true;
        end if
    end for
    return false;
end if
```

The Relation between Nondeterministic Space and Deterministic Space Only Quadratic

Corollary 24 *Let* $f(n) \geq \log n$ *be proper. Then*

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

- Apply Savitch's proof to the configuration graph of the NTM on the input.
- From p. 196, the configuration graph has $O(c^{f(n)})$ nodes; hence each node takes space $O(f(n))$.
- But if we construct explicitly the whole graph before applying Savitch's theorem, we get $O(c^{f(n)})$ space!

The Proof (continued)

- The way out is *not* to generate the graph at all.
- Instead, keep the graph implicit.
- We check for connectedness only when $i = 0$ on p. 204, by examining the input string G.
- There, given configurations x and y, we go over the Turing machine's program to determine if there is an instruction that can turn x into y in one step.[a]

---

[a]Thanks to a lively class discussion on October 15, 2003.

The Proof (concluded)

- The z variable in the algorithm on p. 204 simply runs through all possible valid configurations.
  - Let $z = 0, 1, \ldots, O(c^{f(n)})$.
  - Make sure z is a valid configuration before using it in the recursive calls.[a]
- Each z has length $O(f(n))$ by Eq. (1) on p. 196.

[a] Thanks to a lively class discussion on October 13, 2004.

Implications of Savitch's Theorem

- PSPACE = NPSPACE.
- Nondeterminism is less powerful with respect to space.
- Nondeterminism may be very powerful with respect to time as it is not known if P = NP.

Nondeterministic Space Is Closed under Complement

- Closure under complement is trivially true for deterministic complexity classes (p. 182).
- It is known that[a]

$$\text{coNSPACE}(f(n)) = \text{NSPACE}(f(n)). \tag{2}$$

- So

$$\begin{aligned} \text{coNL} &= \text{NL}, \\ \text{coNPSPACE} &= \text{NPSPACE}. \end{aligned}$$

- But there are still no hints of coNP = NP.

[a] Szelepscényi (1987) and Immerman (1988).

Reductions and Completeness

Degrees of Difficulty

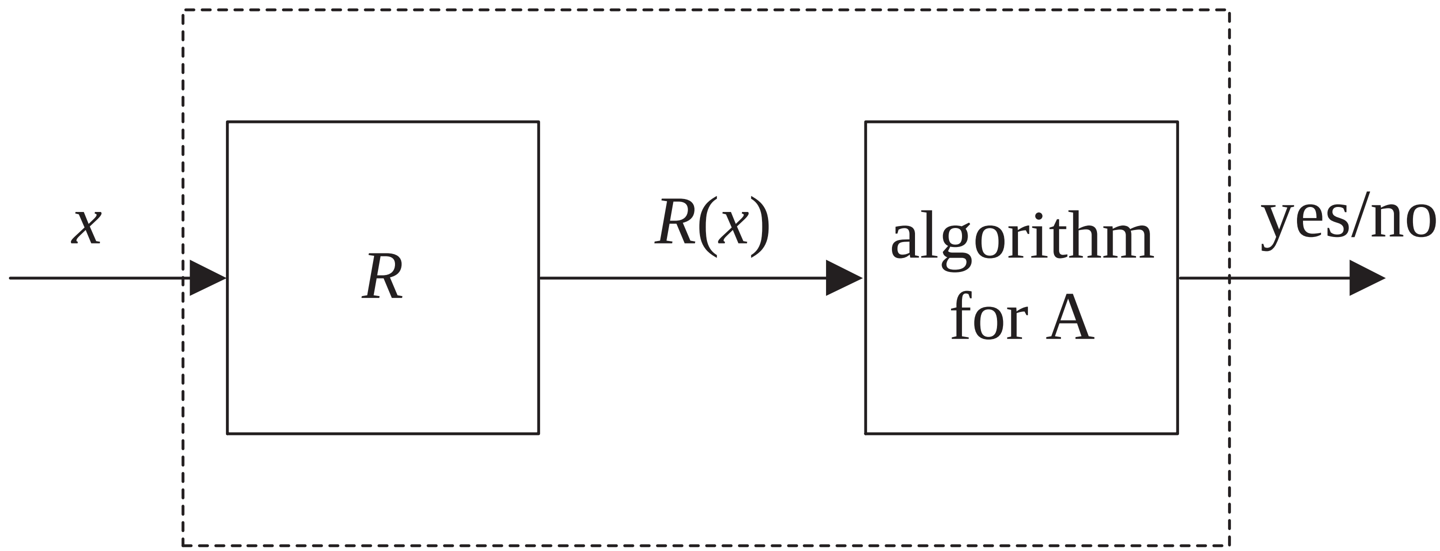
- When is a problem more difficult than another?
- B **reduces to** A if there is a transformation R which for every input x of B yields an equivalent input $R(x)$ of A.
  - The answer to x for B is the same as the answer to $R(x)$ for A.
  - There must be restrictions on the complexity of computing R.
  - Otherwise, $R(x)$ may solve B, defeating the purpose.
    * E.g., $R(x) =$ "yes" if and only if $x \in$ B!

Degrees of Difficulty (concluded)

- We say problem A is at least as hard as problem B if B reduces to A.
- This makes intuitive sense: If A is able to solve your problem B after only a little bit of work of R, then A must be at least as hard.
  - If A is easy to solve, it combined with R (which is also easy) would make B easy to solve, too.[a]
  - If B is hard to solve, A must be hard (if not harder) to solve, too.

[a] Thanks to a lively class discussion on October 13, 2009.

Reduction



Solving problem B by calling the algorithm for problem A *once* and *without* further processing its answer.

Comments[a]

- Suppose B reduces to A via a transformation R.
- The input x is an instance of B.
- The output $R(x)$ is an instance of A.
- $R(x)$ may not span all possible instances of A.[b]
  - Some instances of A may never appear in the range of R.

---

[a]Contributed by Mr. Ming-Feng Tsai (`D92922003`) on October 29, 2003.

[b]$R(x)$ may not be onto; Mr. Alexandr Simak (`D98922040`) on October 13, 2009.

Reduction between Languages

- Language L_1 is **reducible to** L_2 if there is a function R computable by a deterministic TM in space $O(\log n)$.
- Furthermore, for all inputs x, $x \in L_1$ if and only if $R(x) \in L_2$.
- R is said to be a (**Karp**) **reduction** from L_1 to L_2.

Reduction between Languages (concluded)

- Note that by Theorem 22 (p. 193), R runs in polynomial time.
  - In most cases, a polynomial-time R suffices for proofs.
- Suppose R is a reduction from L_1 to L_2.
- Then solving "$R(x) \in L_2$?" is an algorithm for solving "$x \in L_1$?"[a]

[a]But it may not be an optimal one.

A Paradox?

- Degree of difficulty is not defined in terms of *absolute* complexity.
- So a language $B \in \text{TIME}(n^{99})$ may be "easier" than a language $A \in \text{TIME}(n^3)$.
  - This happens when B is reducible to A.
- But isn't this a contradiction if the best algorithm for B requires n^{99} steps?
- That is, how can a problem *requiring* n^{99} steps be reducible to a problem solvable in n^3 steps?

Paradox Resolved

- The so-called contradiction does not hold.
- When we solve the problem "$x \in$ B?" via "$R(x) \in$ A?", we must consider the time spent by $R(x)$ and its length $|\, R(x) \,|$.
- If $|\, R(x) \,| = \Omega(n^{33})$, then answering "$R(x) \in$ A?" takes $\Omega((n^{33})^3) = \Omega(n^{99})$ steps, and there is no contradiction.
- Suppose, on the other hand, that $|\, R(x) \,| = o(n^{33})$.
- Then $R(x)$ must run in time $\Omega(n^{99})$ to make the overall time for answering "$R(x) \in$ A?" take $\Omega(n^{99})$ steps.
- In either case, the contradiction disappears.