Complexity Classes

Complexity Theory

Faculty of Information Technology
Brno University of Technology
Brno, Czech Republic

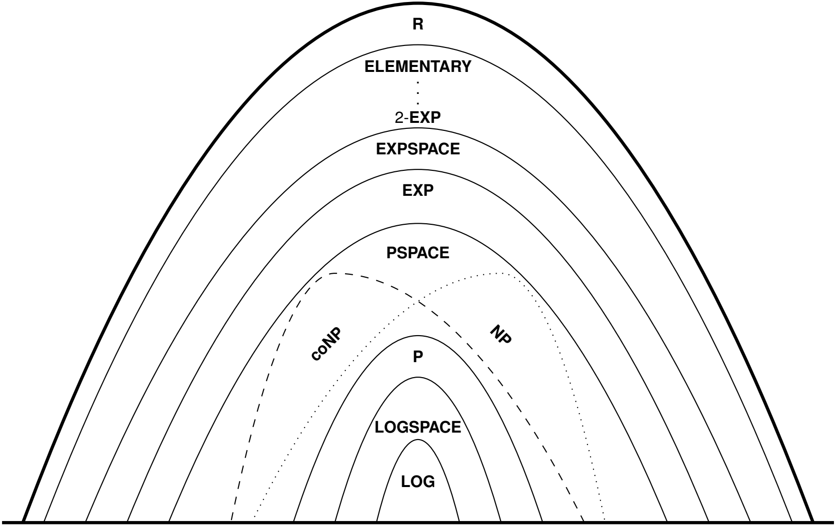
Lukáš Charvát

This material was created with the support of the Czech Ministry of Education, Youth and Sports (project FRVŠ 166/2013/G1).

Complexity Classes – Motivation

- We have studied the relation of a complexity class of a certain type on a function which is used for its definition.
- Now, we will focus on investigating the relation between complexity classes of different types.
- Although a lot effort has been invested in this area in recent years, some problems (such as $P \stackrel{?}{=} NP$) remain unsolved.

Complexity Classes – Motivation



Inclusion of Complexity Classes

- We will show that the following chain of inclusions holds:

$$\mathbf{DLOG} \subseteq \mathbf{NLOG} \subseteq \mathbf{P} \subseteq \mathbf{NP} \subseteq \mathbf{PSPACE} \subseteq \mathbf{EXP} \subseteq \mathbf{NEXP}$$

- To do this, we will prove the statements shown below:
  a) $\mathbf{DSPACE}(f(n)) \subseteq \mathbf{NSPACE}(f(n))$,
  b) $\mathbf{DTIME}(f(n)) \subseteq \mathbf{NTIME}(f(n))$,
  c) $\mathbf{NTIME}(f(n)) \subseteq \bigcup_{c>0} \mathbf{DTIME}(c^{f(n)})$,
  d) $\mathbf{NTIME}(f(n)) \subseteq \mathbf{DSPACE}(f(n))$,
  e) $\mathbf{NSPACE}(f(n)) \subseteq \mathbf{DTIME}(2^{O(f(n))})$ for $f(n) \geq \log n$,
  f) $\mathbf{NSPACE}(f(n)) \subseteq \mathbf{DSPACE}(f(n)^2)$ for $f(n) \geq \log n$ (*Savitch's theorem*).

  where $f(n)$ is a time and space constructible function.

Proof of the Inclusion of Complexity Classes (1/4)

Proof (Parts a, b, c, and d)

- *Parts a)* ($\mathbf{DSPACE}(f(n)) \subseteq \mathbf{NSPACE}(f(n))$) *and b)* ($\mathbf{DTIME}(f(n)) \subseteq \mathbf{NTIME}(f(n))$) *are trivial since each deterministic TM is a special case of a nondeterministic TM.*
- *Part c)* ($\mathbf{NTIME}(f(n)) \subseteq \bigcup_{c>0} \mathbf{DTIME}(c^{f(n)})$) *has been proven in the previous lecture.*
- *The inclusion d)* ($\mathbf{NTIME}(f(n)) \subseteq \mathbf{DSPACE}(f(n))$) *is a consequence of the proof of the proposition* $\mathbf{NTIME}(f(n)) \subseteq \mathbf{DTIME}(2^{f(n)})$. *A nondeterministic TM M_N can make at most $f(n)$ nondeterministic choices. The deterministic TM M_D simulating M_N uses $f(n)$ cells of the tape to represent the choices of M_N and at most $f(n)$ cells for simulating the tape of M_N. M_D simulates the possible runs of M_N one by one.*

Proof of the Inclusion of Complexity Classes (2/4)

Proof (Part e)

- **NSPACE**$(f(n)) \subseteq$ **DTIME**$(2^{O(f(n))})$ *for* $f(n) \geq \log n$
- *We will define the configuration graph* $G(M, w)$ *of an NTM* M *with the input* w*. The vertices of the graph* $G(M, w)$ *are configurations that may occur during a computation of* M *on* w*. There is an edge between the vertices* C_1 *and* C_2 *iff* M *can do a transition from the configuration* C_1 *to* C_2 *in a single step.*
- *We ask whether there is a path from the vertex denoting the initial configuration to the vertex representing the accepting one.*
- *For each input* w*, the number of vertices of* $G(M, w)$ *can be upper-bounded by* $2^{O(\log |w| + f(|w|))}$*.*
- *The already known algorithm REACHABILITY can solve the problem in time* $O(m^2)$ *for graphs with* m *vertices. Therefore, the overall time complexity of a DTM simulating* M *is* $2^{O(\log n + f(n))}$*.*

Proof of the Inclusion of Complexity Classes (3/4)

Proof (Part f)

- $\mathbf{NSPACE}(f(n)) \subseteq \mathbf{DSPACE}(f(n)^2)$ *for* $f(n) \geq \log n$ *(Savitch's th.)*
- *First, we will show that REACHABILITY* $\in \mathbf{DSPACE}(\log^2 n)$.
- *Let G be a graph with n vertices.*
- *Let* x, y *be two distinct vertices from G.*
- *Then, the longest path from* x *to* y *can have at most length n.*
- *We use a DTM to implement the procedure path*(x, y, i) *which returns true iff there exists a path from* x, y *of length at most* 2^i.

```
procedure path(x, y, i)
if (i = 0) then return ((x = y) ∨ ((x,y) ∈ G))
else for all vertices z ∈ G do
    if (path(x, z, i-1) ∧ path(z, y, i-1)) then
        return true
return false
```

Proof of the Inclusion of Complexity Classes (4/4)

Proof (Part f)

- *Recursive calls of path(x, y, i) create a tree of depth i.*
- *For our problem, it suffices to check whether path$(x, y, \lceil \log\ n \rceil)$ holds.*
- *The DTM implementing the path procedure will have to store call stack with at most $\lceil \log\ n \rceil$ triplets of length $3 \cdot \log\ n$.*
- *Therefore, the space complexity of REACHABILITY is $O(\log^2 n)$.*
- *For a NTM M_N we can create a DTM M_D which uses another DTM deciding REACHABILITY for each accepting configuration of M_N.*
- *The number of vertices of $G(M_N, w)$ cannot be bigger than $c^{f(|w|)}$.*
- *Thus, the space complexity of M_D is $O(\log^2 c^{f(n)}) = O(f(n)^2)$.*

Inclusion of Complexity Classes: Conclusion

- We have shown that

$$\mathbf{DLOG} \subseteq \mathbf{NLOG} \subseteq \mathbf{P} \subseteq \mathbf{NP} \subseteq \mathbf{PSPACE} \subseteq \mathbf{EXP} \subseteq \mathbf{NEXP}$$

- We expect all inclusion to be proper but for none of them has this been either proved or refuted yet.
- However, because of Time/Space Hierarchy Theorems (not covered here), it is known that some inclusions need to be proper. It is known that:

$$\mathbf{DLOG} \subset \mathbf{PSPACE}$$

$$\mathbf{P} \subset \mathbf{EXP}$$

$$\mathbf{NP} \subset \mathbf{NEXP}$$

Definition of Complements of Complexity Classes (1/2)

Definition

Let $L \subseteq \Sigma^*$ be a language. The complement of L denoted co–L is the language

$$co\text{–}L = \Sigma^* \setminus L.$$

- The same approach can be extended for decision problems.

Definition

The complement of a decision problem A, denoted A–COMPL, is a problem for which the solution is:

$$\begin{aligned} \textit{yes} &\iff \textit{the solution for } A \textit{ is no} \\ \textit{no} &\iff \textit{the solution for } A \textit{ is yes} \end{aligned}$$

- Formally, *A–COMPL* is not a complement of a language because $A \cup A\text{–}COMPL \neq \Sigma^*$.

Definition of Complements of Complexity Classes (2/2)

Definition

Let $\mathcal{C}$ be a complexity class. The complement of $\mathcal{C}$, denoted co–$\mathcal{C}$, is the complexity class

$$co\text{–}\mathcal{C} = \{co\text{–}L \mid L \in \mathcal{C}\}.$$

- Clearly, deterministic time and space complexity classes are closed under complement:
- each DTM accepting L can be transformed into a DTM accepting $co\text{–}L$ by swapping accepting and rejecting states.

$$\mathbf{DTIME}(f(n)) = co\text{–}\mathbf{DTIME}(f(n))$$

$$\mathbf{DSPACE}(f(n)) = co\text{–}\mathbf{DSPACE}(f(n))$$

Complements of Nondeterministic Complexity Classes

- Nondeterminism introduces an asymmetry in acceptance of a given input by NTM M:

$$w \in L(M) \iff \text{there is an accepting run of } M \text{ on } w,$$
$$w \notin L(M) \iff \text{there is no accepting run of } M \text{ on } w.$$

- Thus, it suffices that one accepting computation exists for the former case but it is required that all computations are rejecting for the latter one.

Nondeterministic Space Complexity Classes (1/8)

Proposition (Immerman-Szelepcsényi Theorem)

$$\mathbf{NSPACE}(f(n)) = co\text{–}\mathbf{NSPACE}(f(n)) \qquad \textit{for } f \geq \log n$$

- We have seen that *REACHABILITY* is in **NLOGSPACE**.
- First, we will demonstrate that the converse of *REACHABILITY*, called *UNREACHABILITY*, is in **NLOGSPACE** too.

Definition

UNREACHABILITY

- *Input: Graph $G = (V, E)$, a pair of vertices $x, y \in V$.*
- *Output: YES if there is no path from x to y, NO otherwise.*

Proposition

$$\textit{UNREACHABILITY} \in \mathbf{NLOGSPACE}$$

Nondeterministic Space Complexity Classes (2/8)

Proof (*UNREACHABILITY* $\in$ **NLOGSPACE**)

- *We will first devise an algorithm that computes the number of reachable nodes in a graph in $O(\log n)$ space,*
  - *and later modify it to solve UNREACHABILITY.*
- *Let $G = (V, E)$ be a graph and $x \in V$.*
- *$S(k)$ is the set of nodes reachable from x in k or less steps,*
  - *$|S(k)|$ is the number of nodes reachable from x in k or less steps.*
- *adjac(v, u) is true iff $v = u \vee (v, u) \in E$, otherwise false.*
- ***procedure*** `numReachable(x, (V, E))`
  - *nondeterministically either **FAIL**s or returns the number of nodes reachable from x,*
  - *we will start with a procedure working in $O(n)$ space and modify it in several steps to use $O(\log n)$ space.*

Nondeterministic Space Complexity Classes (3/8)

Proof (*UNREACHABILITY* $\in$ NLOGSPACE)

```
procedure numReachable(x, (V, E))
|S(0)| := 1
for k := 1 to |V|-1 do
   compute |S(k)| from |S(k-1)|
return |S(|V|-1)|
```

Nondeterministic Space Complexity Classes (4/8)

Proof (*UNREACHABILITY* ∈ NLOGSPACE)

```
procedure numReachable(x, (V, E))
|S(0)| := 1
for k := 1 to |V|-1 do
//compute |S(k)| from |S(k-1)|
  ℓ := 0
  for each node u := 1 to |V| do
    if u ∈ S(k) then ℓ++




  |S(k)| := ℓ
return |S(|V|-1)|
```

Nondeterministic Space Complexity Classes (4/8)

Proof (*UNREACHABILITY* $\in$ NLOGSPACE)

```
procedure numReachable(x, (V, E))
|S(0)| := 1
for k := 1 to |V|-1 do
//compute |S(k)| from |S(k-1)|
  ℓ := 0
  for each node u := 1 to |V| do
    if u ∈ S(k) then ℓ++




  |S(k)| := ℓ
return |S(|V|-1)|
```

Nondeterministic Space Complexity Classes (5/8)

Proof (*UNREACHABILITY* $\in$ NLOGSPACE)

```
procedure numReachable(x, (V, E))
|S(0)| := 1
for k := 1 to |V|-1 do
//compute |S(k)| from |S(k-1)|
  ℓ := 0
  for each node u := 1 to |V| do
//  if u ∈ S(k) then ℓ++
    reply := false
    for each node v := 1 to |V| do
      if v ∈ S(k-1) then
        if adjac(v, u) then reply := true

    if reply then ℓ++
  |S(k)| := ℓ
return |S(|V|-1)|
```

Nondeterministic Space Complexity Classes (5/8)

Proof (*UNREACHABILITY* ∈ **NLOGSPACE**)

```
procedure numReachable(x, (V, E))
|S(0)| := 1
for k := 1 to |V|-1 do
//compute |S(k)| from |S(k-1)|
  ℓ := 0
  for each node u := 1 to |V| do
//   if u ∈ S(k) then ℓ++
    reply := false
    for each node v := 1 to |V| do
      if v ∈ S(k-1) then
        if adjac(v, u) then reply := true

    if reply then ℓ++
  |S(k)| := ℓ
return |S(|V|-1)|
```

Nondeterministic Space Complexity Classes (6/8)

Proof (*UNREACHABILITY* $\in$ **NLOGSPACE**)

```
procedure numReachable(x, (V, E))
|S(0)| := 1
for k := 1 to |V|-1 do
//compute |S(k)| from |S(k-1)|
  ℓ := 0
  for each node u := 1 to |V| do
//  if u ∈ S(k) then ℓ++
    reply := false, m := 0
    for each node v := 1 to |V| do
      if reachNondet(x, v, k-1) then m++    // v ∈ S(k-1)
        if adjac(v, u) then reply := true
    if m < |S(k-1)| then FAIL
    if reply then ℓ++
  |S(k)| := ℓ
return |S(|V|-1)|
```

Nondeterministic Space Complexity Classes (6/8)

Proof (*UNREACHABILITY* ∈ **NLOGSPACE**)

```
procedure numReachable(x, (V, E))
|S(0)| := 1
for k := 1 to |V|-1 do
//compute |S(k)| from |S(k-1)|
  ℓ := 0
  for each node u := 1 to |V| do
//   if u ∈ S(k) then ℓ++
    reply := false, m := 0
    for each node v := 1 to |V| do
      if reachNondet(x, v, k-1) then m++    // v ∈ S(k-1)
        if adjac(v, u) then reply := true
    if m < |S(k-1)| then FAIL
    if reply then ℓ++
  |S(k)| := ℓ
return |S(|V|-1)|
```

guesses `path` from `x` to `v` of length at most `k-1`: returns ***true*** iff `path` is OK, ***false*** if `path` is wrong

if calls to `reachNondet()` did not guess all the paths correctly, terminate the computation

Nondeterministic Space Complexity Classes (6/8)

Proof (*UNREACHABILITY* ∈ NLOGSPACE)

```
procedure numReachable(x, (V, E))
|S(0)| := 1
for k := 1 to |V|-1 do
//compute |S(k)| from |S(k-1)|
  ℓ := 0
  for each node u := 1 to |V| do
//   if u ∈ S(k) then ℓ++
    reply := false, m := 0
    for each node v := 1 to |V| do
      if reachNondet(x, v, k-1) then m++    // v ∈ S(k-1)
        if adjac(v, u) then reply := true
    if m < |S(k-1)| then FAIL
    if reply then ℓ++
  |S(k)| := ℓ
return |S(|V|-1)|
```

Nondeterministic Space Complexity Classes (7/8)

Proof (*UNREACHABILITY* $\in$ NLOGSPACE)

```
procedure reachNondet(x, v, d)
w_0 := x
for p := 1 to d do
  guess a node w_p
  if ¬adjac(w_{p-1}, w_p) then return false
return (w_d = v)
```

- *For UNREACHABILITY*(G, x, y)*, modify* `numReachable()`*:*

```
  ...
    if m < |S(k-1)| then return false
    if reply then ℓ++
    if k = |V|-1 and u = y then return ¬reply
  ...
```

Nondeterministic Space Complexity Classes (8/8)

Proof (*UNREACHABILITY* $\in$ NLOGSPACE)

- *UNREACHABILITY can be implemented using a NTM M that works in space $O(\log n)$.*
- *M keeps binary values of $\texttt{|S(k-1)|}$, ℓ, $\texttt{k}$, $\texttt{u}$, $\texttt{v}$, $\texttt{m}$, $\texttt{reply}$, $\texttt{p}$, $\texttt{w}_p$, $\texttt{w}_{p-1}$ on separate tapes.*
- *Variables are only incremented or compared with each other.*
- *Values of all variables are at most $|V|$, their length at most $\log |V|$.*

Nondeterministic Space Complexity Classes (7/7)

Proposition (Immerman-Szelepcsényi Theorem)

$$\mathbf{NSPACE}(f(n)) = co\text{–}\mathbf{NSPACE}(f(n)) \qquad \textit{for } f \geq \log n$$

Proof

- *Let* $L \in \mathbf{NSPACE}(f(n))$ *be a language decided by a NTM* M_N *and let* w *be its input.*
- *We can construct a DTM* M_D *working in* $\mathbf{NSPACE}(f(n))$ *deciding co–L.*
- M_D *simulates the UNREACHABILITY algorithm for the configuration graph* $G(M, w)$*.*

Corrolary

$$\mathbf{NPSPACE}(f(n)) = co\text{–}\mathbf{NPSPACE}(f(n)), \ldots$$

Nondeterministic Time Complexity Classes (1/2)

- We have seen that

$$\mathbf{NTIME}(f(n)) \subseteq \bigcup_{c>0} \mathbf{DTIME}(c^{f(n)})$$

- Each problem from class **NTIME**$(f(n))$ can be deterministically decided in time $\bigcup_{c>0} \mathbf{DTIME}(c^{f(n)})$.

Corrolary

$$co\text{–}\mathbf{NTIME}(f(n)) \subseteq \bigcup_{c>0} \mathbf{DTIME}(c^{f(n)})$$

Nondeterministic Time Complexity Classes (2/2)

- Relationship between complements of nondeterministic time complexity classes remains unsolved:

$$\mathbf{NTIME}(f(n)) \stackrel{?}{=} co - \mathbf{NTIME}(f(n))$$

Corrolary

$$\mathbf{NP}(f(n)) \stackrel{?}{=} co\text{–}\mathbf{NP}(f(n))$$

Corrolary

$$\mathbf{NEXP}(f(n)) \stackrel{?}{=} co\text{–}\mathbf{EXP}(f(n))$$