CS 577 - Intro to Algorithms

Reductions

Dieter van Melkebeek

November 15, 2022

Outline

Paradigm

Solve a computational problem A using a blackbox for another computational problem B.

Motivation

- Modular design
- NP-completeness

Today

- Notion
- Example where A and B have efficient algorithms
- Examples where A and B have no (known) efficient algorithms: optimization vs search vs decision

Notion

Let A and B be two computational problems.

Definition

A reduction from A to B is an algorithm for A that can make use of a blackbox for B.

Queries

- On a given input x_A of problem A, reduction can make multiple queries x_B to the blackbox for B.
- For a valid query x_B of problem B, the blackbox returns a valid output y_B for problem B on input x_B.
- Often times one query suffices.

Bipartite Matching and Integral Max Flow

A : Bipartite Matching

Input: bipartite graph $G = (V, E)$ where $V = L \sqcup R$ and $E \subseteq L \times R$

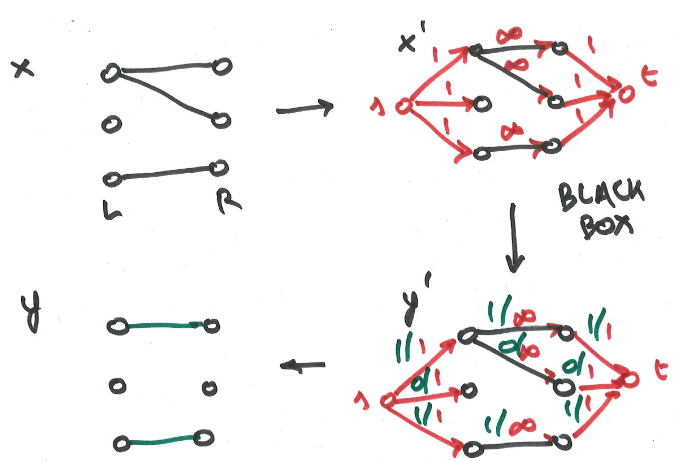
Output: matching M such that $|M|$ is maximized

B : Integral Max Flow

Input: network $N = (V', E', c, s, t)$

Output: integral flow f such that $\nu(f) \doteq f_{\text{out}}(s)$ is maximized

Reduction from Bipartite Matching to Integral Max Flow



Correctness of a Reduction

Definition

On every valid input x_A of A:

- Each query x_B to the blackbox is valid input of B.
- Assuming all queries to the blackbox are answered correctly, the reduction produces a correct output y_A for A on input x_A.

Corollary

Replacing the blackbox for B by a correct algorithm for B yields a correct algorithm for A.

Reduciblity

Definition
$A \leq B$ if there exists a reduction from A to B.

Properties

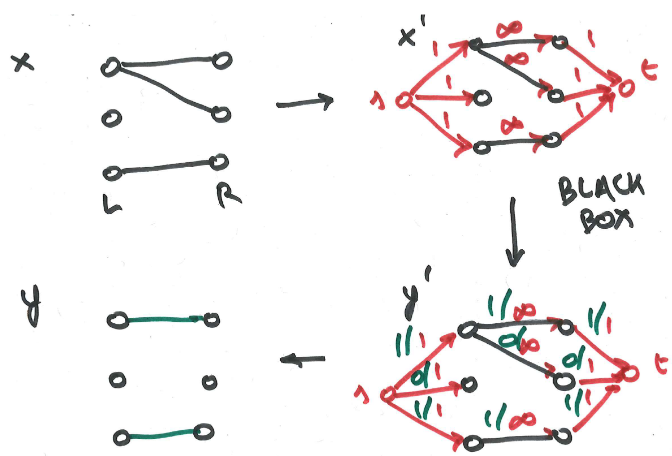
- Reflexive: $A \leq A$
- Not symmetric: Bipartite Matching $\leq$ Halting Problem, but not the other way.
- Transitive: $A \leq B$ and $B \leq C$ implies $A \leq C$.
- If $A \leq B$ and B can be solved algorithmically, then A can be solved algorithmically.

Running Time of a Reduction

Definition
Time to run the reduction assuming blackbox queries are answered instantaneously.

Case of one query

Reduction from Bipartite Matching to Integral Max Flow



Running Time of a Reduction

Definition
Time to run the reduction assuming blackbox queries are answered instantaneously.

Case of one query
Running time of reduction consists of:

- Time to construct out of the input x_A to A the query x_B to B.
- Time to construct out of the answer y_B of B to query x_B, the answer y_A for A on input x_A.
- Not time to compute y_B out of x_B.

Corollary
Suppose reduction from A to B runs in time t. Replacing the blackbox for B by an algorithm for B that runs in time $t_B(n)$ yields an algorithm for A that runs in time $t + t \cdot t_B(t)$.

Polynomial Time

Bit-length
The bit-length of an input x is the number of bits needed to represent x.

- binary strings: length
- numbers: length of the binary representation (finite precision)
- graphs: $O(n^2)$ for adjacency matrix, $O(n + m \log n)$ for adjacency list
- ...

Definition
An algorithm/reduction runs in polynomial time if its running time is $O(n^c)$ for some constant c, where $n \doteq$ bit-length of the input.

Robustness
Notion turns out to be the same for most (but perhaps not all) reasonable input representations and models of computation.

Polynomial–Time Reduciblity

Definition
$A \leq^p B$ if there exists a polynomial-time reduction from A to B.

Properties

- Reflexive: $A \leq^p A$
- Not symmetric
- Transitive: $A \leq^p B$ and $B \leq^p C$ implies $A \leq^p C$.
- If $A \leq^p B$ and B can be solved in polynomial time, then A can be solved in polynomial time.
- If $A \leq^p B$ and A cannot be solved in polynomial time, then B cannot be solved in polynomial time.

Independent Set

Definition
An independent set in a graph $G = (V, E)$ is a subset $S \subseteq V$ such that $E \cap S \times S = \emptyset$.

Example
Valid schedule for unweighted interval scheduling corresponds to independent set in conflict graph.

Computational problems
- Optimization: solution or value
- Search
- Decision

Independent Set – problem specifications

OptSol
Input: graph G
Output: independent set S of G such that $|S|$ is maximized

OptVal
Input: graph G
Output: size of largest independent set of G

Search
Input: graph G, $k \in \mathbb{N}$
Output: independent set S of G such that $|S| \geq k$, or report that no such set exists

Decision
Input: graph G, $k \in \mathbb{N}$
Output: whether independent set S with $|S| \geq k$ exists in G

Independent Set – Decision $\leq^p$ Search $\leq^p$ OptSol

- Decision $\leq^p$ Search

```
if Search(G, k) = "no solution" then
    return "no"
else
    return "yes"
```

- Search $\leq^p$ OptSol

```
I ← OptSol(G)
if |I| ≥ k then
    return I
else
    return "no solution"
```

Independent Set – Decision $\leq^p$ OptVal $\leq^p$ OptSol

- Decision $\leq^p$ OptVal

```
if k ≤ OptVal(G) then
    return "yes"
else
    return "no"
```

- OptVal $\leq^p$ OptSol

```
return |OptSol(G)|
```

Independent Set – Optimization $\leq^p$ Search $\leq^p$ Decision

- OptSol $\leq^p$ Search
  - Linear search for maximum size

```
k ← 0
while Search(G, k) ≠ "no solution" do
    k ← k + 1
return Search(G, k)
```

  - Binary search reduces number of queries from $O(|V|)$ to $O(\log |V|)$.
- Search $\leq^p$ Decision: next slides
- Corollaries:
  - OptSol $\leq^p$ Decision $\leq^p$ OptVal
  - OptVal $\leq^p$ Decision

Independent Set – Search $\leq^p$ Decision

- Vertex v has to be in *every* independent set of size at least k $\Leftrightarrow$ Decision$(G - \{v\}, k) =$ "no"
- Reluctant approach

```
if Decision(G, k) = "no" then
    return "no solution"
I ← V
for each v ∈ V do
    if Decision(G|_{I\{v}}, k) = "yes" then
        I ← I \ {v}
return I
```

- Considering vertices in lexicographical order results in independent set of size at least k with the lexicographically first characteristic vector.

Independent Set – Search $\leq^p$ Decision

- Vertex v can be in *some* independent set of size at least k $\Leftrightarrow$ Decision$(G - (\{v\} \cup G(v)), k-1) =$ "yes"
- Eager approach

  if Decision$(G, k) =$ "no" **then**
      return "no solution"
  $I \leftarrow \emptyset$; $S \leftarrow V$
  while $S \neq \emptyset$ **do**
      pick $v \in S$; $S \leftarrow S \setminus \{v\}$
      if Decision$(G|_{S \setminus G(v)}, k-1) =$ "yes" **then**
          $I \leftarrow I \cup \{v\}$
          $S \leftarrow S \setminus G(v)$
          $k \leftarrow k - 1$
  return I

- Considering vertices in lexicographical order results in independent set of size at least k with the lexicographically last characteristic vector.

CS 577 - Intro to Algorithms

Reductions

Dieter van Melkebeek

November 17, 2022

Outline

Definition
A reduction from computational problem A to computational problem B is an algorithm for A that uses a blackbox for B.

Running time
Time to run reduction discounting time for blackbox.

Notation
$A \leq^p B$: A reduces to B in polynomial time.

Example polynomial-time reductions
- Bipartite Matching $\leq^p$ Integral Max Flow
- Between different versions of Independent Set: decision, search, optimal value, optimal solution
- Independent Set vs Satisfiability

Boolean Formulas

Definition
- Base case: variables $x_1, x_2, \ldots$
- Constructors:
  - Conjunction (AND): $\wedge$
  - Disjunction (inclusive OR): $\vee$
  - Negation: $\neg$ also denoted as $\overline{}$

Example
$[(x_1 \vee x_2 \vee \neg x_3) \wedge \neg x_1] \vee x_4 \equiv [(x_1 \vee x_2 \vee \overline{x_3}) \wedge \overline{x_1}] \vee x_4$

Restricted types
- Literal: variable x, negated variable $\overline{x}$
- Clause: disjunction of literals
- Conjunctive normal form (CNF): conjunction of clauses

Conjunctive Normal Form

$$(x_1 \vee \overline{x_2} \vee x_4) \wedge (\overline{x_1} \vee x_3)$$

Theorem
For every $f : \{0,1\}^k \to \{0,1\}$ there exists a CNF-formula φ such that for every $x = x_1 \ldots x_n \in \{0,1\}^n$

$$f(x_1, \ldots, x_k) = 1 \Leftrightarrow \varphi(x_1, \ldots, x_k).$$

Proof

$$\begin{aligned} f(x_1, \ldots, x_k) = 1 &\Leftrightarrow \wedge_{a\,:\,f(a)=0} (x \neq a) \\ &\Leftrightarrow \wedge_{a\,:\,f(a)=0} \vee_{i=1}^{k} (x_i \neq a_i) \\ &\Leftrightarrow \wedge_{a\,:\,f(a)=0} \vee_{i=1}^{k} x_i^{1-a_i} \end{aligned}$$

where $x^0 \doteq \overline{x}$ and $x^1 \doteq x$.

Note: Size of φ can be exponential in k.

Satisfiability

Search version
Input: Boolean formula φ

Output: satisfying assignment of φ: setting of the variables to true/false that makes φ evaluate to true; or report that no such setting exists

Decision version
Input: Boolean formula φ

Output: whether φ has a satisfying assignment

Restricted problems
- CNF-SAT: φ is CNF
- k-SAT for fixed $k \in \mathbb{N}$: φ is k-CNF, i.e., CNF with each clause containing at most k literals

CNF-SAT $\leq^p$ Independent Set

A : CNF-SAT Decision

Input: CNF-formula φ: $\wedge_{j=1}^{m} C_j$ where $C_j = \vee_{r=1}^{k_j} \ell_{jr}$ and $\ell_{jr} \in \{x_1, \overline{x_1}, \ldots x_n, \overline{x_n}\}$

Output: whether φ has a satisfying assignment

B : Independent Set Decision

Input: graph $G = (V, E)$, $k \in \mathbb{N}$

Output: whether G has an independent set of size at least k

Typical properties of reduction $A \leq^p B$

- Makes a single query: (G, k)
- Mapping reduction: translation of CNF-SAT instance φ into Independent Set instance (G, k) with same decision, i.e., φ is satisfiable $\Leftrightarrow$ G has independent set of size at least k.
- Gadget reduction

CNF-SAT $\leq^p$ Independent Set – variable gadgets

$\wedge_{j=1}^{m} C_j$ with $C_j = \vee_{r=1}^{k_j} \ell_{jr}$ and $\ell_{jr} \in \{x_1, \overline{x_1}, \ldots x_n, \overline{x_n}\} \rightarrow (G, k)$

Construction

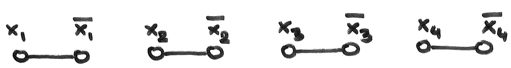
For each variable x_i, include two new vertices, one labeled x_i and the other $\overline{x_i}$, and include the edge $(x_i, \overline{x_i})$.

Properties

- Maximum size of independent set is n.
- Bijection between
  - independent sets of maximum size and
  - assignments to variables $x_1, x_2, \ldots, x_n$.

CNF-SAT $\leq^p$ Independent Set – variable gadgets

$(x_1 \vee \overline{x_2} \vee x_4) \wedge (\overline{x_1} \vee x_3)$



CNF-SAT $\leq^p$ Independent Set – clause gadgets

$\wedge_{j=1}^{m} C_j$ with $C_j = \vee_{r=1}^{k_j} \ell_{jr}$ and $\ell_{jr} \in \{x_1, \overline{x_1}, \ldots x_n, \overline{x_n}\} \rightarrow (G, k)$

Construction

For each clause C_j for $j \in [m]$, include a clique (complete graph) on k_j new vertices, where k_j = number of literals of C_j. Label each vertex of the clique with a unique literal of C_j.

Properties

- Maximum size of independent set is m.
- Bijection between
  - independent sets of maximum size and
  - choices of literal in each clause C_j for $j \in [m]$.

CNF-SAT $\leq^p$ Independent Set – clause gadgets

$(x_1 \vee \overline{x_2} \vee x_4) \wedge (\overline{x_1} \vee x_3)$

CNF-SAT $\leq^p$ Independent Set – connections

Construction of G

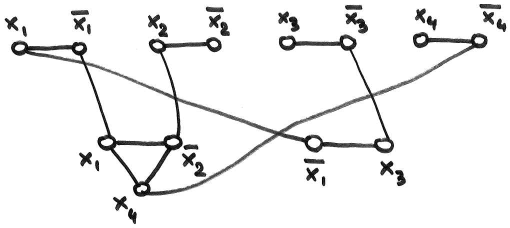
- Disjoint union of all variable gadgets and clause gadgets.
- For each variable gadget vertex labeled ℓ, and each clause gadget vertex labeled $\overline{\ell}$, include edge between them.

Properties

- Max independent set size in G is at most $n + m$.
- Independent set of size n in variable part can be extended with vertex in gadget of clause $C_j \Leftrightarrow$ assignment satisfies C_j.
- Max independent set size in G is at least $k \doteq n + m$ $\Leftrightarrow \varphi$ has a satisfying assignment
- (G, k) can be constructed in polynomial time.
- Note: Bijection between
  - independent sets of size $n + m$ in G and
  - satisfying assignments to $x_1, x_2, \ldots, x_n$ combined with choices of satisfying literal in each clause C_j for $j \in [m]$.

CNF-SAT $\leq^p$ Independent Set – connections

$$(x_1 \vee \overline{x_2} \vee x_4) \wedge (\overline{x_1} \vee x_3)$$



Independent Set $\leq^p$ CNF-SAT

A : Independent Set Decision

Input: graph $G = (V, E)$, $k \in \mathbb{N}$

Output: whether G has an independent set of size at least k

B : CNF-SAT Decision

Input: CNF-formula φ

Output: whether φ has a satisfying assignment

Typical properties of reduction $A \leq^p B$

- Makes a single query: φ
- Mapping reduction: translation of Independent Set instance (G, k) into CNF-SAT instance φ with same decision, i.e., G has independent set of size at least $k \Leftrightarrow \varphi$ is satisfiable.
- Gadget reduction

Independent Set $\leq^p$ CNF-SAT $\quad (G, k) \to \varphi$

Modeling independent set

- Introduce variable x_v for each vertex $v \in V$.
- x_v indicates whether v belongs to the independent set.

Enforcing independence condition

- For every edge $e = (u, v)$ include clause $\overline{x_u} \vee \overline{x_v}$.

Enforcing size requirement

- Introduce new variable for each bit of binary representation of $\sum_{v \in V} x_v$.
- Include clauses to enforce correct values for the new variables.
- Involves introduction of auxiliary variables. [on board]
- Include additional clauses and auxiliary variables to enforce $\sum_{v \in V} x_v \geq k$ using binary representation.