Discrete Optimization

MA2827

Fondements de l'optimisation discrète

https://project.inria.fr/2015ma2827/

Material from M. Pawan Kumar, E. Demaine

Outline

- Preliminaries
  - Random Access Machine (RAM)
  - Polynomial-time Algorithm
  - Decision Problems
- P, NP, and NP-Complete Problems

Random Access Machine

- Given an array f with elements = 0, 1 strings
- RAM executes a set of instructions
- Each instruction can
  - Read entries from prescribed positions
  - Perform arithmetic operation on read entries
  - Write answers to prescribed positions

Random Access Machine

- Given an array f with elements = 0, 1 strings
- Finite set of variables z_0, z_1, … , z_k
- Initially, z_i = 0 and f contains input
- Instructions numbered 0, 1, …, t
- Variable z_0 stores the instruction to execute

Random Access Machine

- Given an array f with elements = 0, 1 strings
- Finite set of variables z_0, z_1, … , z_k
- Initially, $z_i = 0$ and f contains input
- Instructions numbered 0, 1, …, t
- Stop if $z_0 > t$

Random Access Machine

- Given an array f with elements = 0, 1 strings
- Finite set of variables z_0, z_1, … , z_k
- Initially, $z_i = 0$ and f contains input
- Read instruction
  - $z_i := f(z_j)$

Random Access Machine

- Given an array f with elements = 0, 1 strings
- Finite set of variables z_0, z_1, … , z_k
- Initially, $z_i = 0$ and f contains input
- Write instruction
  - $f(z_i) := z_j$

Random Access Machine

- Given an array f with elements = 0, 1 strings
- Finite set of variables $z_0, z_1, \ldots, z_k$
- Initially, $z_i = 0$ and f contains input
- Add instruction
  - $z_i := z_j + z_k$

Random Access Machine

- Given an array f with elements = 0, 1 strings

- Finite set of variables $z_0, z_1, \dots, z_k$

- Initially, $z_i = 0$ and f contains input

- Subtract instruction
  - $z_i := z_j - z_k$

Random Access Machine

- Given an array f with elements = 0, 1 strings
- Finite set of variables $z_0, z_1, \ldots, z_k$
- Initially, $z_i = 0$ and f contains input
- Multiply instruction
  - $z_i := z_j z_k$

Random Access Machine

- Given an array f with elements = 0, 1 strings

- Finite set of variables z_0, z_1, … , z_k

- Initially, $z_i = 0$ and f contains input

- Divide instruction
  - $z_i := z_j / z_k$

Random Access Machine

- Given an array f with elements = 0, 1 strings
- Finite set of variables $z_0, z_1, \ldots, z_k$
- Initially, $z_i = 0$ and f contains input
- Increment instruction
  - $z_i := z_i + 1$

Random Access Machine

- Given an array f with elements = 0, 1 strings
- Finite set of variables $z_0, z_1, \ldots, z_k$
- Initially, $z_i = 0$ and f contains input
- Binarize instruction
  - $z_i := 1$, if $z_i > 0$
  - $z_i := 0$, otherwise

Random Access Machine

$z_1 = 0$

$z_2 = 0$

$z_3 = 0$

$z_4 = 0$

Array f

10	3	0

→

- $z_2 := f(z_1)$
- $z_1 := z_1+1$
- $z_3 := f(z_1)$
- $z_4 := z_2 + z_3$
- $z_1 := z_1+1$
- $f(z_1) := z_4$

Random Access Machine

$z_1 = 0$

$z_2 = 10$

$z_3 = 0$

$z_4 = 0$

Array f

10	3	0

- $z_2 := f(z_1)$
- $z_1 := z_1+1$
- $z_3 := f(z_1)$
- $z_4 := z_2 + z_3$
- $z_1 := z_1+1$
- $f(z_1) := z_4$

Random Access Machine

$z_1 = 0$

$z_2 = 10$

$z_3 = 0$

$z_4 = 0$

Array f

10	3	0

- $z_2 := f(z_1)$
- → $z_1 := z_1+1$
- $z_3 := f(z_1)$
- $z_4 := z_2 + z_3$
- $z_1 := z_1+1$
- $f(z_1) := z_4$

Random Access Machine

$z_1 = 1$

$z_2 = 10$

$z_3 = 0$

$z_4 = 0$

Array f

10	3	0

- $z_2 := f(z_1)$
- → $z_1 := z_1+1$
- $z_3 := f(z_1)$
- $z_4 := z_2 + z_3$
- $z_1 := z_1+1$
- $f(z_1) := z_4$

Random Access Machine

$z_1 = 1$

$z_2 = 10$

$z_3 = 0$

$z_4 = 0$

Array f

10	3	0

- $z_2 := f(z_1)$
- $z_1 := z_1+1$
- $z_3 := f(z_1)$
- $z_4 := z_2 + z_3$
- $z_1 := z_1+1$
- $f(z_1) := z_4$

Random Access Machine

$z_1 = 1$

$z_2 = 10$

$z_3 = 3$

$z_4 = 0$

Array f

10	3	0

- $z_2 := f(z_1)$
- $z_1 := z_1+1$
- → $z_3 := f(z_1)$
- $z_4 := z_2 + z_3$
- $z_1 := z_1+1$
- $f(z_1) := z_4$

Random Access Machine

$z_1 = 1$

$z_2 = 10$

$z_3 = 3$

$z_4 = 0$

Array f

10	3	0

- $z_2 := f(z_1)$
- $z_1 := z_1+1$
- $z_3 := f(z_1)$
- → $z_4 := z_2 + z_3$
- $z_1 := z_1+1$
- $f(z_1) := z_4$

Random Access Machine

$z_1 = 1$

$z_2 = 10$

$z_3 = 3$

$z_4 = 13$

Array f

10	3	0

- $z_2 := f(z_1)$
- $z_1 := z_1+1$
- $z_3 := f(z_1)$
- → $z_4 := z_2 + z_3$
- $z_1 := z_1+1$
- $f(z_1) := z_4$

Random Access Machine

$z_1 = 1$

$z_2 = 10$

$z_3 = 3$

$z_4 = 13$

Array f

10	3	0

- $z_2 := f(z_1)$
- $z_1 := z_1+1$
- $z_3 := f(z_1)$
- $z_4 := z_2 + z_3$
- → $z_1 := z_1+1$
- $f(z_1) := z_4$

Random Access Machine

$z_1 = 2$

$z_2 = 10$

$z_3 = 3$

$z_4 = 13$

Array f

10	3	0

- $z_2 := f(z_1)$
- $z_1 := z_1+1$
- $z_3 := f(z_1)$
- $z_4 := z_2 + z_3$
- → $z_1 := z_1+1$
- $f(z_1) := z_4$

Random Access Machine

$z_1 = 2$

$z_2 = 10$

$z_3 = 3$

$z_4 = 13$

Array f

10	3	0

- $z_2 := f(z_1)$
- $z_1 := z_1+1$
- $z_3 := f(z_1)$
- $z_4 := z_2 + z_3$
- $z_1 := z_1+1$
- $f(z_1) := z_4$

Random Access Machine

$z_1 = 2$

$z_2 = 10$

$z_3 = 3$

$z_4 = 13$

Array f

10	3	13

- $z_2 := f(z_1)$
- $z_1 := z_1+1$
- $z_3 := f(z_1)$
- $z_4 := z_2 + z_3$
- $z_1 := z_1+1$
- $f(z_1) := z_4$

Outline

- Preliminaries
  - Random Access Machine (RAM)
  - **Polynomial-time Algorithm**
  - Decision Problems

- P, NP, and NP-Complete Problems

Polynomial Time Algorithm

- Input size = Number of bits b
- Polynomial time algorithm
  - Number of instructions = t
  - t is bounded by a polynomial in b
  - Good algorithm
  - Efficient algorithm

Outline

- Preliminaries
  - Random Access Machine (RAM)
  - Polynomial-time Algorithm
  - **Decision Problems (Answered by ‘yes’ or ‘no’)**
- P, NP, and NP-Complete Problems
- Reduction

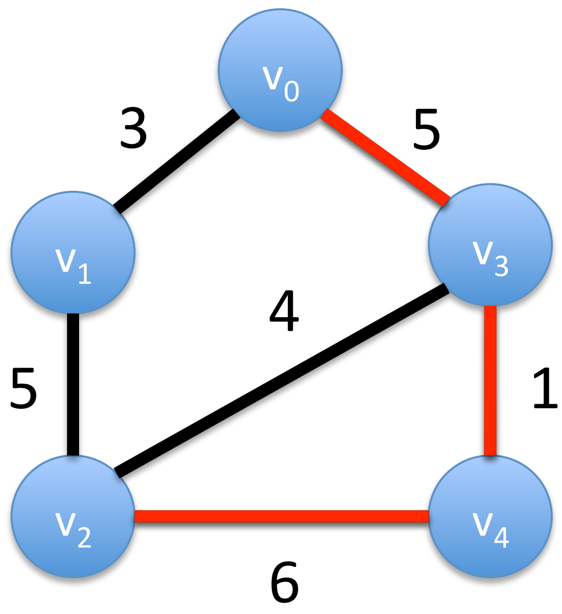
Decision Problem

- Finite set Σ called alphabet of size ≥ 2
  - {0,1}
  - {a,b,c,d,e,…,x,y,z}
- Set Σ* of all finite length strings (called words)
  - 0, 1, 00, 01, 10, 11, 000,….
  - discrete, optimization
- Size of word size(w) = number of letters
  - size(00) = 2
  - size(discrete) = 8

Decision Problem

- Problem Π is a subset of Σ^*
  - All words with the answer "yes"
- Informal problem
  - Given input word $x \in \Sigma^*$, does $x \in \Pi$?
- Polynomial-time solvable problem Π
  - There exists a polynomial-time algorithm for the informal problem
  - Polynomial in size(x)

Shortest Path



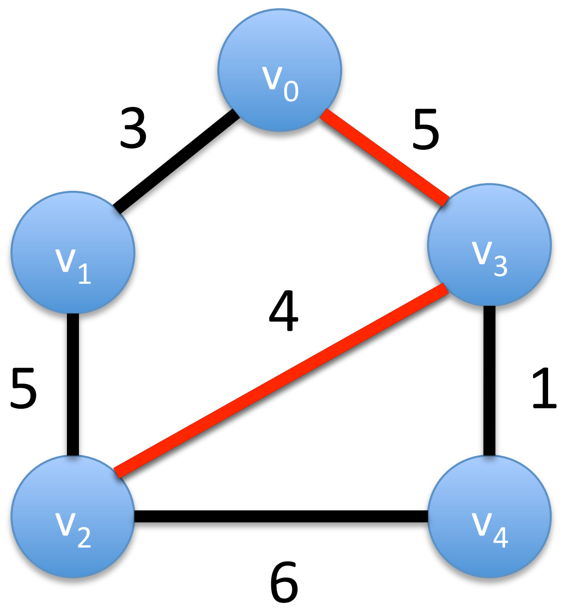
$v_0, v_1, v_2 \ldots v_n \in \Sigma$

$\{ \in \Sigma$ $\} \in \Sigma$ $, \in \Sigma$

$0,1,2, \ldots N \in \Sigma$

$\{v_0,v_1,v_2,v_3,v_4,v_5\}$,
$\{\{v_0,v_1\},\{v_0,v_3\},\{v_1,v_2\},\{v_2,v_3\},\{v_2,v_4\},\{v_3,v_4\}\}$, $\in \Sigma^*$
$\{3,5,5,4,6,1\}$,
$\{v_0,v_3,v_4,v_2\}$

Shortest Path



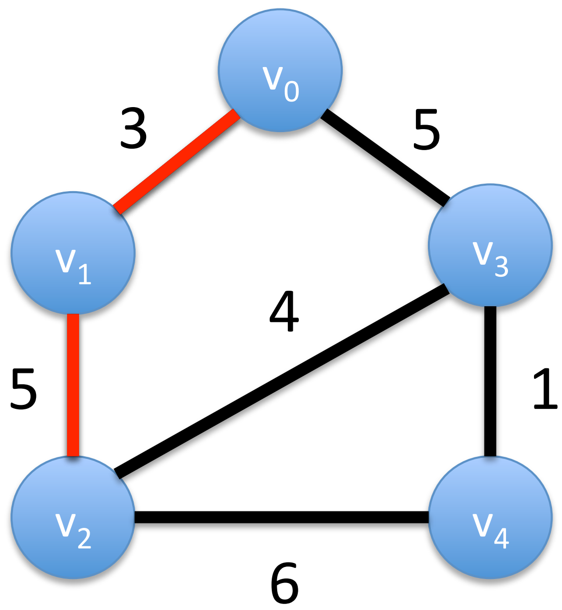
$v_0, v_1, v_2 \ldots v_n \in \Sigma$

$\{ \in \Sigma$ $\} \in \Sigma$ $, \in \Sigma$

$0, 1, 2, \ldots N \in \Sigma$

$\{v_0,v_1,v_2,v_3,v_4,v_5\},$
$\{\{v_0,v_1\},\{v_0,v_3\},\{v_1,v_2\},\{v_2,v_3\},\{v_2,v_4\},\{v_3,v_4\}\},$ $\in \Sigma^*$
$\{3,5,5,4,6,1\},$
$\{v_0,v_3,v_2\}$

Shortest Path



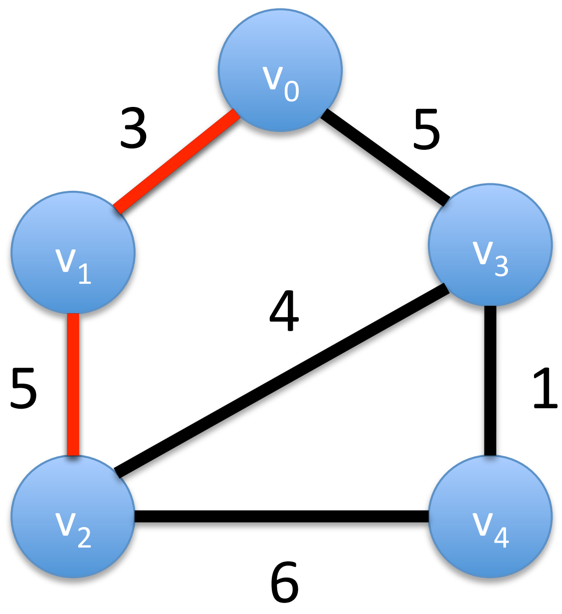
$v_0, v_1, v_2 \ldots v_n \in \Sigma$

$\{ \in \Sigma$ $\} \in \Sigma$ $, \in \Sigma$

$0,1,2, \ldots N \in \Sigma$

$\{v_0,v_1,v_2,v_3,v_4,v_5\},$
$\{\{v_0,v_1\},\{v_0,v_3\},\{v_1,v_2\},\{v_2,v_3\},\{v_2,v_4\},\{v_3,v_4\}\},$ $\in \Sigma^*$
$\{3,5,5,4,6,1\},$
$\{v_0,v_1,v_2\}$

Shortest Path



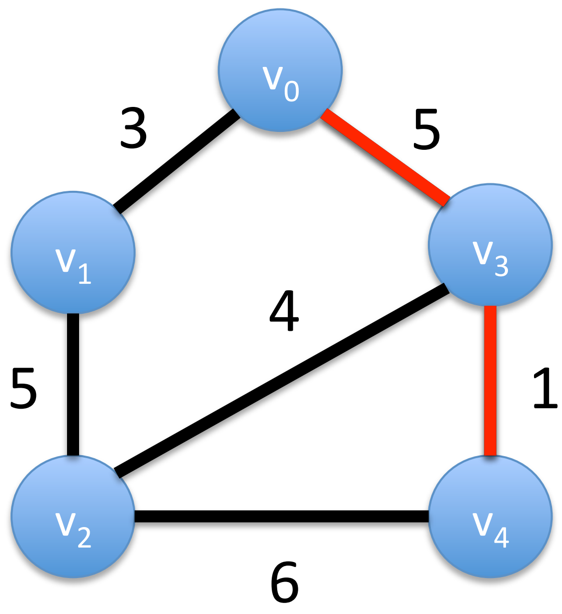
$v_0, v_1, v_2 \ldots v_n \in \Sigma$

$\{ \in \Sigma$ $\} \in \Sigma$ $, \in \Sigma$

$0,1,2, \ldots N \in \Sigma$

$\{v_0,v_1,v_2,v_3,v_4,v_5\},$
$\{\{v_0,v_1\},\{v_0,v_3\},\{v_1,v_2\},\{v_2,v_3\},\{v_2,v_4\},\{v_3,v_4\}\},$ $\in \Pi$
$\{3,5,5,4,6,1\},$
$\{v_0,v_1,v_2\}$

Shortest Path



$v_0, v_1, v_2 \ldots v_n \in \Sigma$

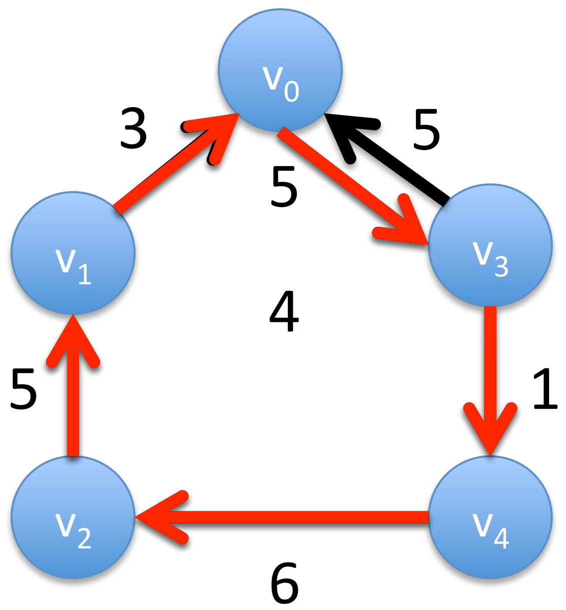
$\{ \in \Sigma \qquad \} \in \Sigma \qquad , \in \Sigma$

$0, 1, 2, \ldots N \in \Sigma$

Given graph G and path P, is P a shortest path in G?

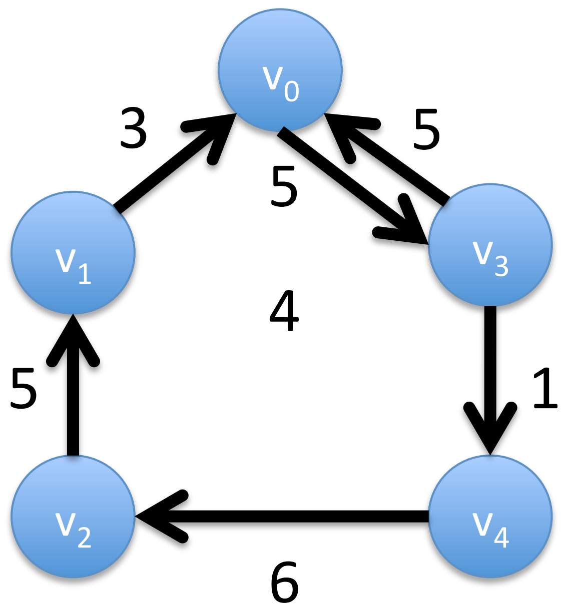
$\{v_0, v_1, v_2, v_3, v_4, v_5\},$
$\{\{v_0, v_1\}, \{v_0, v_3\}, \{v_1, v_2\}, \{v_2, v_3\}, \{v_2, v_4\}, \{v_3, v_4\}\},$ $\quad \in \Pi$
$\{3, 5, 5, 4, 6, 1\},$
$\{v_0, v_3, v_4\}$

Hamiltonian Circuit



Circuit consisting of all vertices

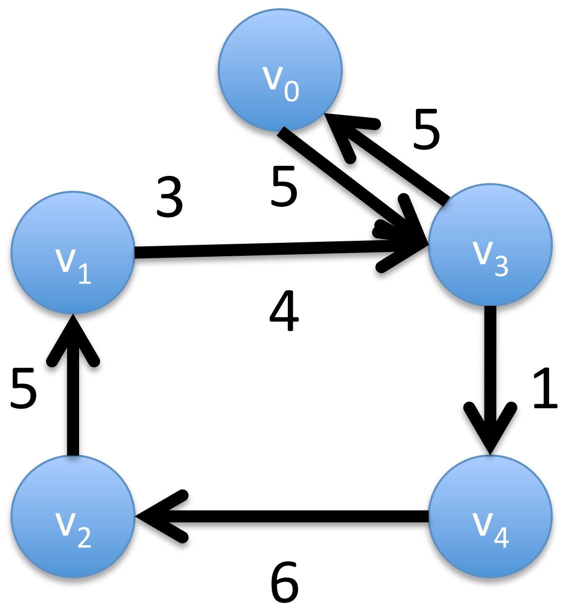
Hamiltonian Graph



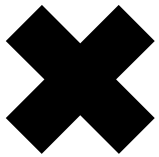
Has a Hamiltonian Circuit

✔

Hamiltonian Graph

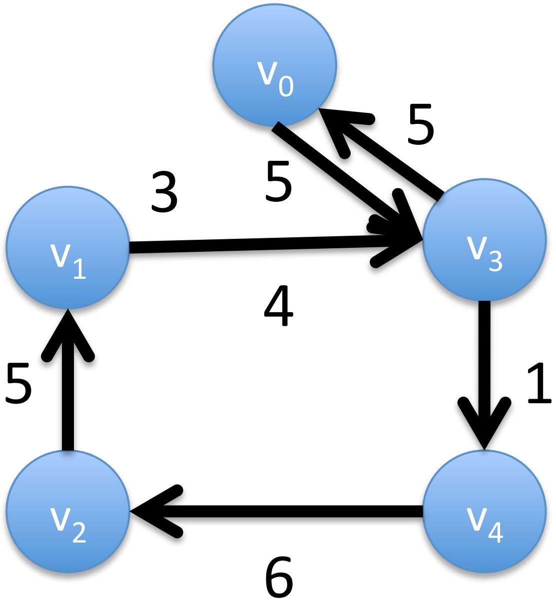


Has a Hamiltonian Circuit



Given graph G, is the graph Hamiltonian?

Hamiltonian Graph



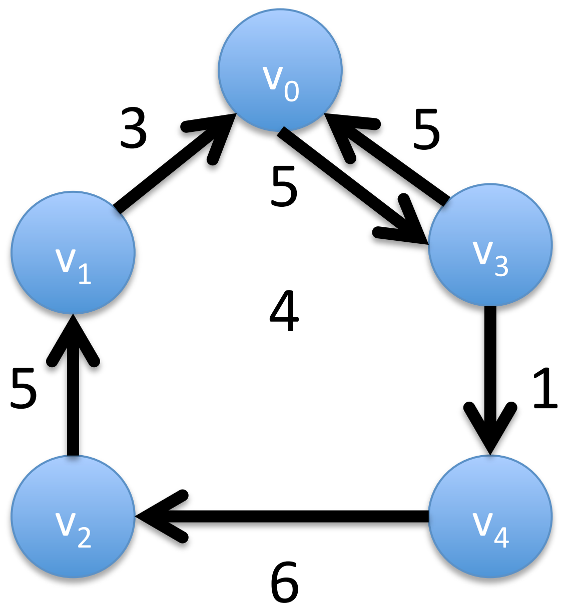
$v_0, v_1, v_2 \ldots v_n \in \Sigma$

$\{ \in \Sigma \qquad \} \in \Sigma \qquad , \in \Sigma$

$0, 1, 2, \ldots N \in \Sigma$

$\{v_0,v_1,v_2,v_3,v_4,v_5\},$
$\{\{v_0,v_3\},\{v_1,v_3\},\{v_2,v_1\},\{v_3,v_0\},\{v_3,v_4\},\{v_4,v_2\}\}, \qquad \in \Sigma^*$

Hamiltonian Graph



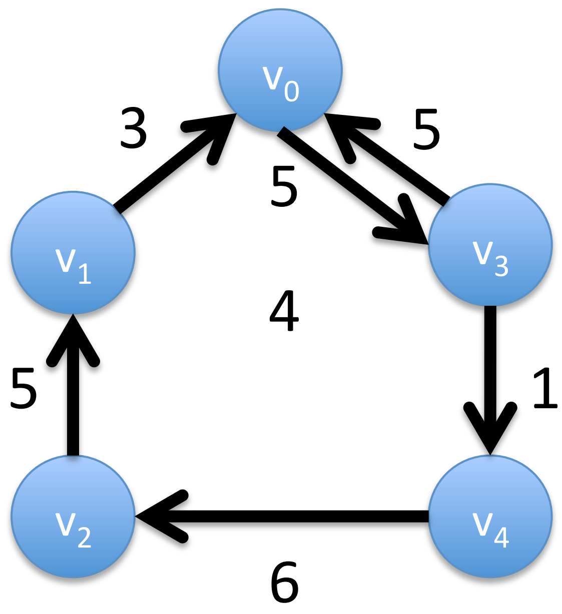
$v_0, v_1, v_2 \ldots v_n \in \Sigma$

$\{ \in \Sigma$ $\} \in \Sigma$ $, \in \Sigma$

$0, 1, 2, \ldots N \in \Sigma$

$\{v_0,v_1,v_2,v_3,v_4,v_5\},$
$\{\{v_0,v_3\},\{v_1,v_0\},\{v_2,v_1\},\{v_3,v_0\},\{v_3,v_4\},\{v_4,v_2\}\},$ $\in \Sigma^*$

Hamiltonian Graph



$v_0, v_1, v_2 \ldots v_n \in \Sigma$

$\{ \in \Sigma$ $\} \in \Sigma$ $, \in \Sigma$

$0, 1, 2, \ldots N \in \Sigma$

$\{v_0,v_1,v_2,v_3,v_4,v_5\},$
$\{\{v_0,v_3\},\{v_1,v_0\},\{v_2,v_1\},\{v_3,v_0\},\{v_3,v_4\},\{v_4,v_2\}\},$ $\in \Pi$

Outline

- Preliminaries
- **P, NP, and NP-Complete Problems**

P

- Polynomial-time solvable problem Π
  - There exists a polynomial-time algorithm that decides whether $x \in \Sigma^*$ belongs to Π or not.
  - Polynomial in size(x)
- P = {all Π, Π is polynomial time solvable}
- For example
  - Shortest path with +ve lengths $\in$ P
  - Shortest path with no –ve length circuit $\in$ P

NP

- $\Pi \in NP$
  - There exists a problem $\Pi' \in P$
  - There exists a polynomial p
  - There exists an x, size(x) $\leq$ p(size(w)) such that
  - $w \in \Pi$ if and only if $wx \in \Pi'$

- Polynomial-time checkable 'certificate'

- For example
  - Hamiltonian graph problem $\in NP$

NP

- Relationship between P and NP?

- P is a subset of NP
  - "P = NP or not" is an open problem
  - One of Clay Institute's Millennium Prize problems

- For example
  - Shortest path with +ve lengths $\in$ NP
  - Shortest path with no –ve length circuit $\in$ NP

NP-Complete

- Hardest problems in NP
  - All problems in NP can be *reduced* to an NP-complete problem

- Π is reducible to Λ
  - There exists a polynomial time algorithm
  - Given w, returns x
  - $w \in \Pi$ if and only if $x \in \Lambda$

- If $\Lambda \in P$ then $\Pi \in P$

NP-Complete

- Hardest problems in NP
  - All problems in NP can be *reduced* to an NP-complete problem

- Π is reducible to Λ
  - There exists a polynomial time algorithm
  - Given w, returns x
  - $w \in \Pi$ if and only if $x \in \Lambda$

- If $\Lambda \in NP$ then $\Pi \in NP$

NP-Complete

- Hardest problems in NP
  - All problems in NP can be *reduced* to an NP-complete problem

- Π is reducible to Λ
  - There exists a polynomial time algorithm
  - Given w, returns x
  - w ∈ Π if and only if x ∈ Λ

- If Λ ∈ NP-complete and Λ ∈ P, then P = NP

P, NP, EXP, etc.

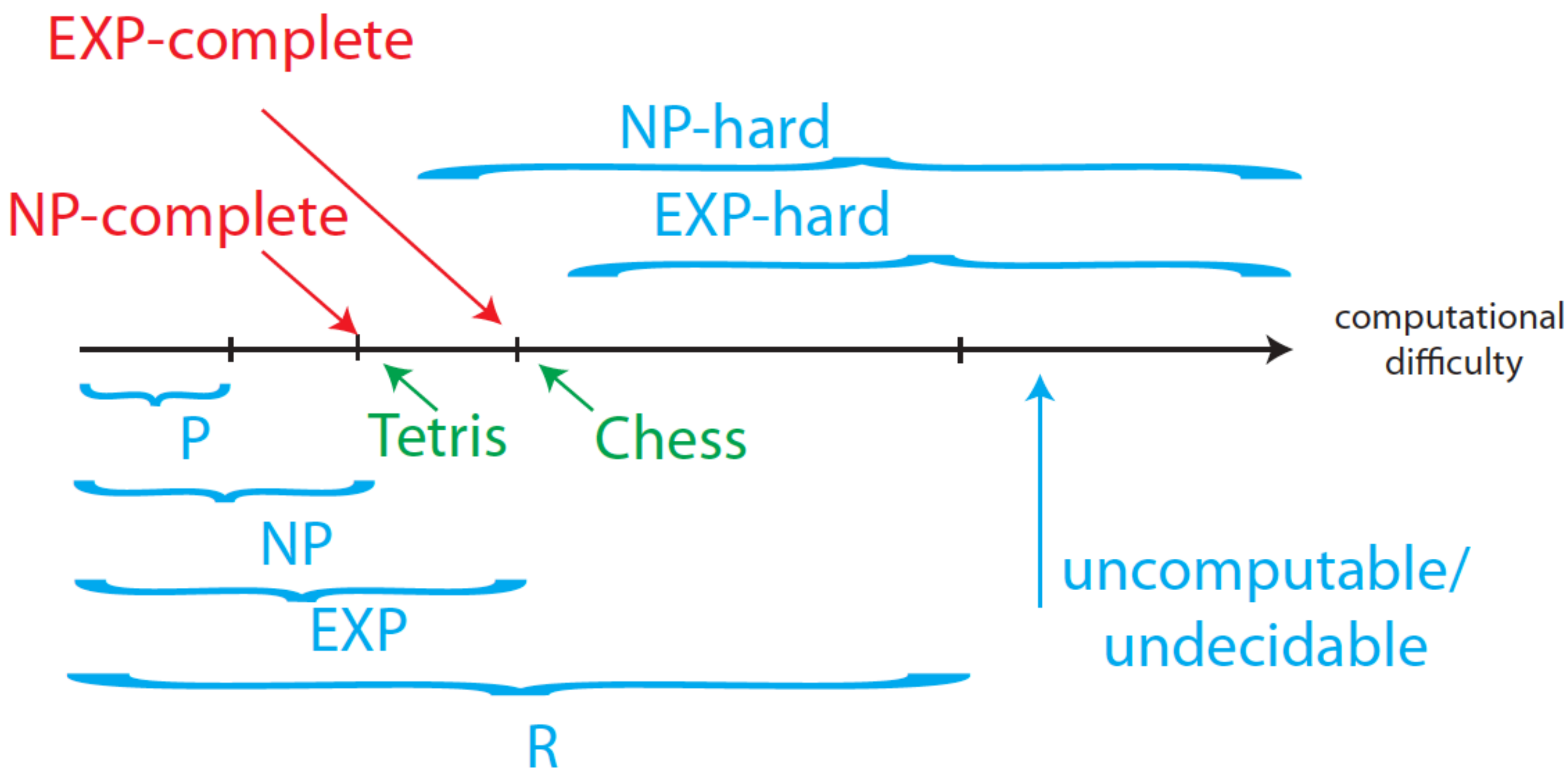


Image Courtesy of E. Demaine

Outline

- Reduction
  - "SAT" is reducible to "3-SAT"
  - "3-SAT" is reducible to "Partition"
  - "Partition" is reducible to "Hamiltonian Path"
- NP-hard Problems
- NP-completeness of SAT

Boolean Expression

- Alphabet Σ containing *variables* $x_1, x_2, \ldots, x_n$
- And special symbols (,), $\wedge$, $\vee$, ~
- And not containing 0 and 1
- A variable is a Boolean expression

Boolean Expression

- Alphabet Σ containing *variables* $x_1, x_2, \ldots, x_n$
- And special symbols (,), $\wedge$, $\vee$, ~
- And not containing 0 and 1
- If v and w are Boolean expressions then
  - $(v \wedge w)$ is a Boolean expression

Boolean Expression

- Alphabet Σ containing *variables* $x_1, x_2, \ldots, x_n$
- And special symbols (,), $\wedge$, $\vee$, ~
- And not containing 0 and 1
- If v and w are Boolean expressions then
  - (v $\vee$ w) is a Boolean expression

Boolean Expression

- Alphabet Σ containing *variables* $x_1, x_2, \ldots, x_n$
- And special symbols (,), $\wedge$, $\vee$, ~
- And not containing 0 and 1
- If v and w are Boolean expressions then
  - ~v is a Boolean expression
  - ~w is a Boolean expression

Boolean Expression

- Alphabet Σ containing *variables* $x_1, x_2, \ldots, x_n$
- And special symbols (,), $\wedge$, $\vee$, ~
- And not containing 0 and 1
- For example, $f(x_1, x_2, \ldots, x_n)$
  - $((x_2 \wedge x_3) \vee \sim(x_3 \vee x_5) \wedge x_2) \vee \sim(x_2 \wedge x_5)$
  - $\sim(\sim x_2 \vee \sim x_3) \wedge \sim x_1$

Satisfiability

- $f(x_1,x_2,\ldots,x_n)$ is satisfiable if
  - there exists an assignment $x_1 = \alpha_1, \ldots, x_n = \alpha_n$
  - where $\alpha_i \in \{0,1\}$
  - such that $f(x_1,x_2,\ldots,x_n) = 1$

- The following identities hold
  - $0 \wedge 0 = 0$, $0 \wedge 1 = 0$, $1 \wedge 0 = 0$, $1 \wedge 1 = 1$
  - $0 \vee 0 = 0$, $0 \vee 1 = 1$, $1 \vee 0 = 1$, $1 \vee 1 = 1$
  - $\sim 0 = 1$, $\sim 1 = 0$
  - $(0) = 0$, $(1) = 1$

SAT

- Given the alphabet Σ
- Given the identities for 0 and 1
- SAT is a subset of Σ* that is satisfiable
- Informal Problem
  - Given a Boolean expression w, is w satisfiable

3-SAT

- A special case of SAT

- Given variables $x_1, x_2, \ldots, x_n \in \Sigma$
  - B_1 consists of $x_1, \sim x_1, \ldots, x_n, \sim x_n$
  - B_2 consists of $(w_1 \vee \ldots \vee w_k)$, $w_i \in B_1$, $1 \le k \le 3$
  - B_3 consists of $w_1 \wedge w_2 \wedge \ldots \wedge w_m$, $w_i \in B_2$
  - e.g., $(x_1 \vee \sim x_2 \vee x_3) \wedge (x_2 \vee \sim x_3 \vee x_4) \wedge (\sim x_1 \vee \sim x_2)$

- 3-SAT is the subset of B_3 that is satisfiable

SAT is reducible to 3-SAT

- $x_1 = x_2 \lor x_3$
  - $(x_1 \lor \sim x_2) \land (x_1 \lor \sim x_3) \land (\sim x_1 \lor x_2 \lor x_3)$
- $x_1 = x_2 \land x_3$
  - $(\sim x_1 \lor x_2) \land (\sim x_1 \lor x_3) \land (x_1 \lor \sim x_2 \lor \sim x_3)$
- $x_1 = \sim x_2$
  - $(x_1 \lor x_2) \land (\sim x_1 \lor \sim x_2)$

SAT is reducible to 3-SAT

- Example on the board

$$(x_1 \wedge x_2) \vee \sim((\sim x_1 \vee x_3) \wedge x_4 \wedge \sim x_5) \wedge \sim x_2$$

SAT is reducible to 3-SAT

- Example on the board

$$(x_1 \land x_2) \lor \sim((\sim x_1 \lor x_3) \land x_4 \land \sim x_5) \land \sim x_2$$

- (Using new variables + truth table)

Outline

- Reduction
  - "SAT" is reducible to "3-SAT"
  - **"3-SAT" is reducible to "Partition"**
  - "Partition" is reducible to "Hamiltonian Path"
- NP-hard Problems
- NP-completeness of SAT

3-SAT

- A special case of SAT

- Given variables $x_1, x_2, \ldots, x_n \in \Sigma$
  - B_1 consists of $x_1, \sim x_1, \ldots, x_n, \sim x_n$
  - B_2 consists of $(w_1 \lor \ldots \lor w_k)$, $w_i \in B_1$, $1 \le k \le 3$
  - B_3 consists of $w_1 \land w_2 \land \ldots \land w_m$, $w_i \in B_2$
  - e.g., $(x_1 \lor \sim x_2 \lor x_3) \land (x_2 \lor \sim x_3 \lor x_4) \land (\sim x_1 \lor \sim x_2)$

- 3-SAT is the subset of B_3 that is satisfiable

Partition

- A finite set X
- Partition of X is a collection of subsets
  - Mutually exclusive
  - Collectively exhaustive
- For example, X = {a,b,c,d,e,f}
  - {{a,b},{c},{d,e,f}} is a partition
  - {{a,b},{a,c},{d,e,f}} is not a partition
  - {{a,b},{c},{d,e}} is not a partition

Partition

- A finite set X
- Partition of X is a collection of subsets
  - Mutually exclusive
  - Collectively exhaustive
- Problem: Given collection of subsets C
  - Does C contain a partition of X?
  - Or not?

3-SAT is reducible to Partition

- $f = w_1 \wedge w_2 \wedge \ldots w_m$

- Bipartite undirected graph with $V = V_1 \cup V_2$
  - V_1 are variables $x_1, x_2, \ldots, x_n$
  - V_2 are words $w_1, w_2, \ldots, w_m$

- Edges $E = E_1 \cup E_2$
  - $E_1 = \{(w_i, x_j)\}$, $x_j \in w_i$
  - $E_2 = \{(w_i, x_j)\}$, $\sim x_j \in w_i$

3-SAT is reducible to Partition

- Collection C_1 of sets $\{w_i\}$ U E_i
  - E_i is non-empty
  - E_i is a subset of edge set incident with w_i

- Collection C_2 of sets $\{x_j\}$ U E_j and $\{x_j\}$ U E'_j
  - E_j is the set of all edges in E_1 incident with x_j
  - E'_j is the set of all edges in E_2 incident with x_j

- f is satisfiable iff C_1 U C_2 contains a partition

Outline

- Reduction
  - "SAT" is reducible to "3-SAT" (review)
  - "3-SAT" is reducible to "Partition"
  - **"Partition" is reducible to "Hamiltonian Path"**
- NP-hard Problems
- NP-completeness of SAT

Partition

- A finite set X

- Partition of X is a collection of subsets
  - Mutually exclusive
  - Collectively exhaustive

- Problem: Given collection of subsets C
  - Does C contain a partition of X?
  - Or not?

Hamiltonian Path

- Digraph D = (V, A)

- Path P is Hamiltonian if
  - It traverses each vertex in V
  - All vertices in the path are distinct

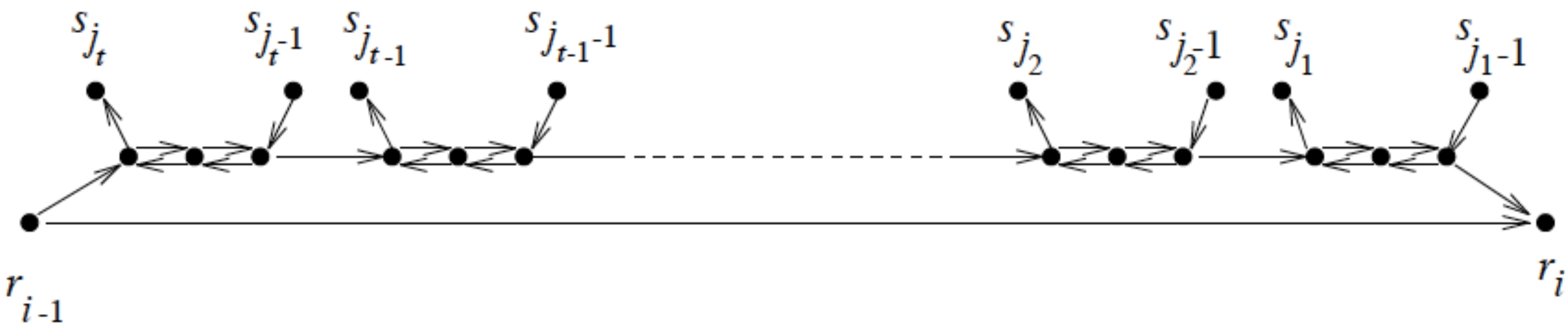
- Problem: Given D
  - Does it contain a Hamiltonian Path?
  - Or not?

Connection to Shortest Path?

Partition is reducible to Hamiltonian

- Partition Problem: Set X, Collection C
  - $X = \{1,2,\dots,k\}$
  - $C = \{C_1,C_2,\dots,C_m\}$

- Let $C_i = \{j_1,\dots,j_t\}$

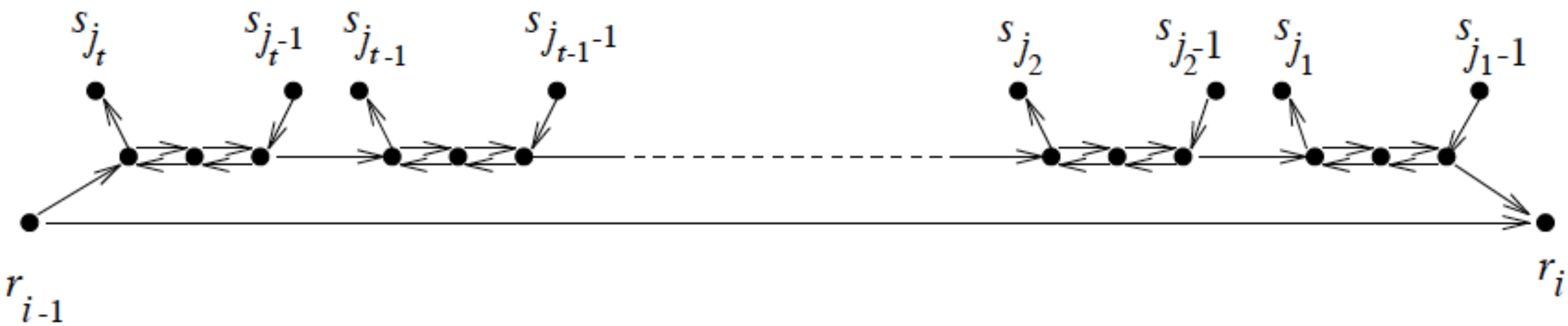


Introduce r_0 and s_0. Connect r_m to s_0.

Partition is reducible to Hamiltonian

- Partition Problem: Set X, Collection C
  - $X = \{1,2,\ldots,k\}$
  - $C = \{C_1, C_2, \ldots, C_m\}$

- Let $C_i = \{j_1, \ldots, j_t\}$

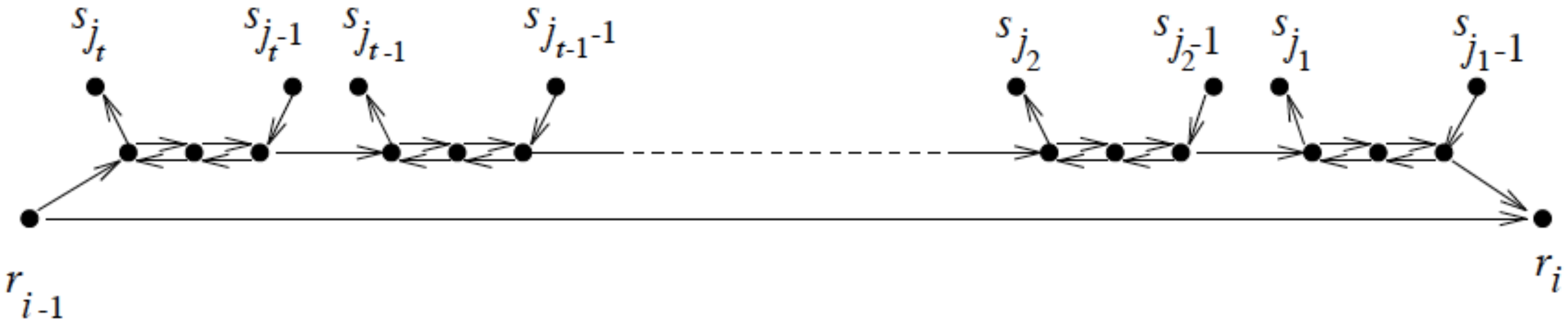


C has a partition iff G has Hamiltonian r_0-s_k path

Partition is reducible to Hamiltonian

- Partition Problem: Set X, Collection C
  - $X = \{1,2,\ldots,k\}$
  - $C = \{C_1,C_2,\ldots,C_m\}$

- Let $C_i = \{j_1,\ldots,j_t\}$



Left as Exercise !!

Outline

- Reduction
- **NP-hard Problems**
- NP-completeness of SAT

NP-hard

- Need not be in NP
  - No polynomial-time checkable certificate
  - Not even a decision problem (e.g. optimization)

- At least as hard as NP-complete problems

- $\Lambda \in$ NP-hard
  - There exists $\Pi \in$ NP-complete
  - Π is reducible to Λ

NP-hard

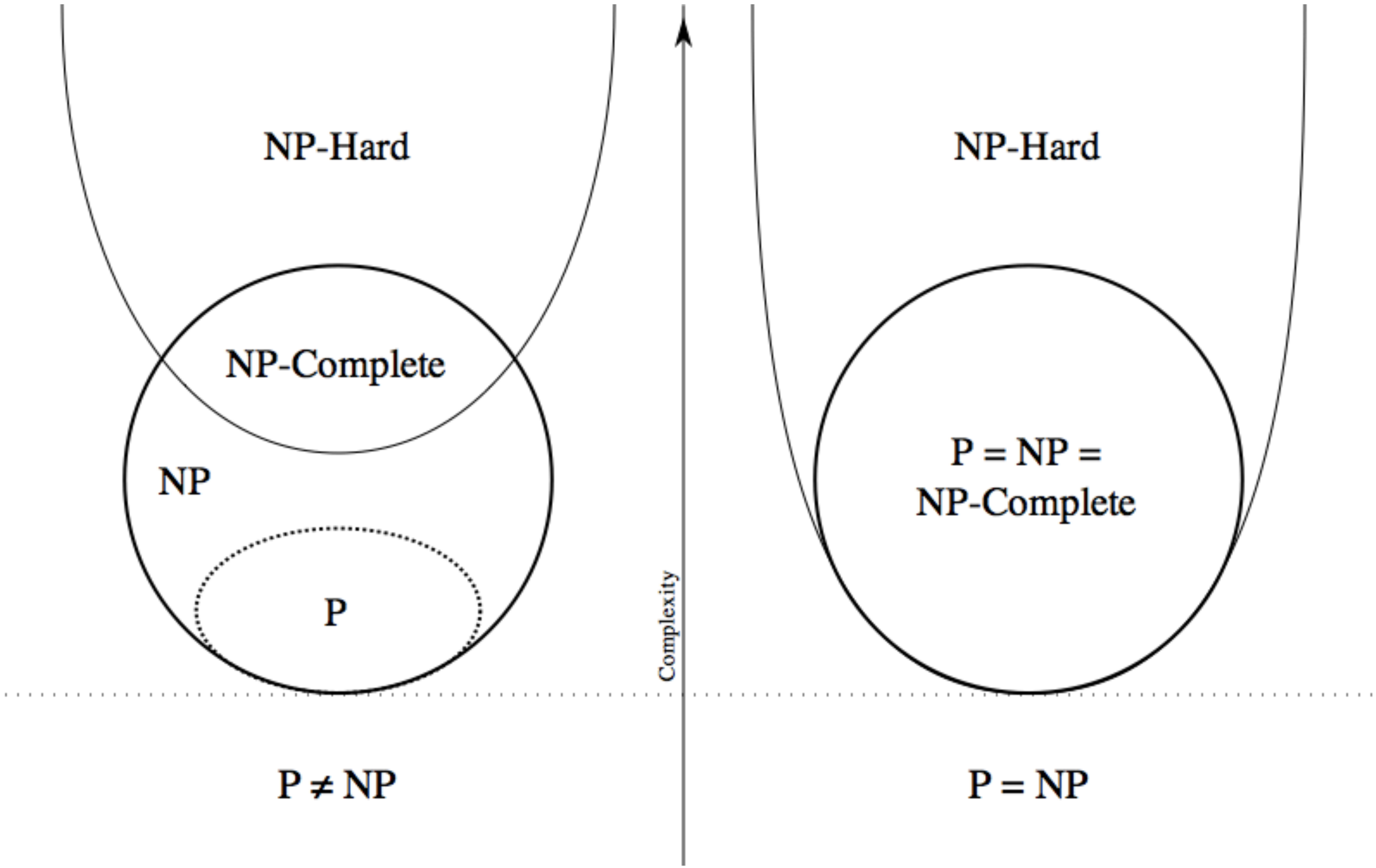


Image Courtesy of Wikipedia

Examples of NP-complete problems

- 3-Partition: given n integers, can you divide them into triples of equal sum?
- Travelling salesman problem
- Tetris
- Minesweeper, Sudoku
- SAT
- Knapsack (pseudopoly, **not** poly)
- …

Outline

- Reduction
- NP-hard Problems
- **NP-completeness of SAT (todo)**