APPROXIMABILITY

- Approximation Algorithms
- Approximation and complexity
- Nonapproximability results

(C. Papadimitriou: *Computational complexity*, Chapter 13, 299–322)

1. Approximation Algorithms

- Once **NP**-completeness of a problem has been established, techniques for solving the problem only approximatively are usually explored.
- When dealing with optimization problems, often heuristic (search) algorithms are used.
- Such algorithms are valuable in practice even if usually nothing can be proved about their worst-case (or expected) performance.
- In some (fortunate) cases, the solutions returned by a polynomial-time heurictic algorithm are guaranteed to be "not too far from the optimum"

© 2007 TKK, Laboratory for Theoretical Computer Science

Approximation Algorithms

Definition. In an optimization problem there is an infinite set of instance such that for each instance, there is a set of *feasible solutions* $F(x)$ and for each such solution $s \in F(x)$, we have a positive integer cost $c(s)$. The task is to find a feasible solution having the optimum cost defined as $\mathrm{OPT}(x) = \min_{s \in F(x)} c(s)$ (or $\max_{s \in F(x)} c(s)$ if A is a maximization problem).

Let M be an algorithm which given any instance x returns a feasible solution $M(x) \in F(x)$. We say that M is an *ε-approximation algorithm*, where $\varepsilon \geq 0$, iff for all inputs x,

$$\frac{|c(M(x)) - \mathrm{OPT}(x)|}{\max\{\mathrm{OPT}(x), c(M(x))\}} \leq \varepsilon.$$

© 2007 TKK, Laboratory for Theoretical Computer Science

Approximation Algorithms

- Note that ε-approximation means that the relative error is at most ε
- For a minimization problem

$$\frac{|c(M(x)) - \mathrm{OPT}(x)|}{\max\{\mathrm{OPT}(x), c(M(x))\}} = \frac{c(M(x)) - \mathrm{OPT}(x)}{c(M(x))} \leq \varepsilon$$

  and hence, $c(M(x)) \leq \frac{1}{1-\varepsilon}\mathrm{OPT}(x)$.

- For a maximization problem

$$\frac{|c(M(x)) - \mathrm{OPT}(x)|}{\max\{\mathrm{OPT}(x), c(M(x))\}} = \frac{\mathrm{OPT}(x) - c(M(x))}{\mathrm{OPT}(x)} \leq \varepsilon$$

  and hence, $c(M(x)) \geq (1-\varepsilon)\mathrm{OPT}(x)$.

© 2007 TKK, Laboratory for Theoretical Computer Science

Approximation Thresholds

- For an optimization problem A we are interested in determining the smallest ε for which there is a polynomial-time ε-approximation algorithms for A.
- Sometimes no such smallest ε exists but there are approximization algorithms that achieve arbitrarily small error ratios.
- The *approximation threshold* of A is the greatest lower bound (**glb**) of all $\varepsilon > 0$ for which A has a polynomial-time ε-approximation algorithm.
- This quantity ranges from 0 (arbitrarily closer approximation is possible) to 1 (essentially no approximation is possible).
- If $\mathbf{P} = \mathbf{NP}$, then for all optimization problems in $\mathbf{NP}$, the approximation threshold is zero.

Node Cover

- NODE COVER is a minimization problem where we seek the smallest set of nodes $C \subseteq V$ in a graph $G = (V, E)$ such that for each edge in E at least one of its endpoints is in C.
- What is a plausible heurictic for obtaining a "good" node cover?
- A first try: If a node v has high degree, then it is probably a good idea to add it to the cover.
- The resulting "greedy" algorithm:
  Start with $C = \emptyset$;
  While there are still edges left in G
  choose a node with the largest degree, delete it (and related edges) from G and add it to C.
- This is not an ε-approximation algorithm for any $\varepsilon < 1$ (in the worst-case its error ratio grows as $\log n$ where n is the number of nodes in the graph).

Node Cover

- To get an approximation algorithm for NODE COVER a less "greedy" approach needs to be taken such as:
  Start with $C = \emptyset$;
  While there are still edges left in G
  choose any edge $[u, v]$, add both u and v to C and delete them from G.
- How far off the optimum can C be?
  - C contains $\frac{1}{2}|C|$ edges of G (no two of which share a node).
  - Also the optimum cover must contain at least one node from each such edge.
  - Hence, $\mathrm{OPT}(G) \geq \frac{1}{2}|C|$ and thus $\frac{|C| - \mathrm{OPT}(G)}{|C|} \leq \frac{1}{2}$.

Theorem. The approximation threshold of NODE COVER is at most $\frac{1}{2}$.

Maximum Satisfiability

- Consider first the k-MAXGSAT problem (maximum generalized satisfiability): we are given a set of Boolean expressions $\Phi = \{\phi_1, \ldots, \phi_m\}$ in n variables where each expression is a general Boolean expression involving at most k of the n variables ($k > 0$ is fixed constant). The task is to find a truth assignment that satisfies the most expressions
- A successful approximation algorithm is based on choosing for a variable always the truth value that maximizes the expected number of satisfied expressions.

Maximum Satisfiability

The expected number of satisfied expressions:

- Suppose we pick one of the 2^n truth assignments at random. How many expressions in Φ should we expect to satisfy?
- Each expression $\phi_i \in \Phi$ involves k Boolean variables.
- We can easily calculate the number t_i of truth assignments (out of 2^k truth assignments) that satisfy ϕ_i (as k is a constant).
- Thus, a random truth assignment will satisfy ϕ_i with probability $p(\phi_i) = \frac{t_i}{2^k}$
- The expected number of satisfied expressions is then $p(\Phi) = \sum_{i=1}^{m} p(\phi_i)$

Maximum Satisfiability

- If we set x_1 to **true** in all expressions of Φ, a set of expressions $\Phi[x_1 = \mathbf{true}]$ involving variables $x_2, \ldots, x_n$ results. We can calculate again $p(\Phi[x_1 = \mathbf{true}])$ (and for $\Phi[x_1 = \mathbf{false}]$ similarly). Now it holds that

$$p(\Phi) = \frac{1}{2}(p(\Phi[x_1 = \mathbf{true}]) + p(\Phi[x_1 = \mathbf{false}]))$$

- Hence, if we modify Φ by setting x_1 equal to the truth value t that yields the largest $p(\Phi[x_1 = t])$, we end up with an expression set with expectation at least as large as the original.
- The approximation algorithm:
  Set $\Phi' = \Phi$ and then for $i = 1$ to n
  compute $p(\Phi'[x_i = \mathbf{true}])$ and $p(\Phi'[x_i = \mathbf{false}])$; choose the truth value t that yields the largest $p(\Phi'[x_i = t])$; set $\Phi' = \Phi'[x_1 = t]$.

Maximum Satisfiability

- In the end, all variables have been given values and all expressions are either **true** or **false** but we know that at least $p(\Phi)$ have been satisfied.
- The optimum is at most the number of expressions that can be individually satisfied ($p(\phi_i) > 0$).

$$\frac{\mathrm{OPT}(\Phi)-c(M(\Phi))}{\mathrm{OPT}(\Phi)} = 1 - \frac{c(M(\Phi))}{\mathrm{OPT}(\Phi)} \leq 1 - \frac{p(\Phi)}{\mathrm{OPT}(\Phi)} \leq$$
$$1 - \frac{l p(\phi_i)}{\mathrm{OPT}(\Phi)} \leq 1 - \frac{l p(\phi_i)}{l} = 1 - p(\phi_i)$$

  where l is the number of expressions ϕ_j with $p(\phi_j) > 0$ and $p(\phi_i)$ is the smallest positive probability.
- For every satisfiable expression ϕ_i involving k variables $p(\phi_i)$ is at least 2^{-k}.
- Hence, the approximation threshold for k-MAXGSAT is at most $1 - 2^{-k}$.

MAXSAT

- In MAXSAT the input is a set of clauses and the probability of satisfaction is at least $\frac{1}{2}$ and $\varepsilon = \frac{1}{2}$.
- If we restrict the clauses to have at least k distinct literals, the probability that a random truth assignment satisfies a clause is $1 - 2^{-k}$ and $\varepsilon = 2^{-k}$.

Theorem. The approximation threshold for k-MAXGSAT is at most $1 - 2^{-k}$.
The approximation threshold for MAXSAT is at most $\frac{1}{2}$
and when each clause has at least k distinct literals, the approximation threshold is at most 2^{-k}.

Maximum Cut

- In MAX CUT we want to partition the nodes of a graph $G = (V, E)$ into two sets S and $V - S$ such that there are as many edges as possible between S and $V - S$.
- An approximation algorithm of MAX CUT based on *local improvement*:
  Start from any partition of the nodes of G and repeat the following step: If the cut can be made larger by adding a single node to S or by deleting a single node from S, then do so. If no such improvement is possible, stop and return the cut thus obtained.
- Such local improvement algorithms can be developed for just about any optimization problem.
- Sometimes such algorithms work well in practice but usually very little can be proved about their performance.
- MAX CUT is an exception:

Theorem. The approximation threshold for MAX CUT is at most $\frac{1}{2}$.

The Traveling Salesperson Problem

- TSP cannot be approximated!

  Theorem. Unless $\mathbf{P} = \mathbf{NP}$, the approximation threshold for TSP is one.
- If all distances are either 1 or 2, there is a polynomial-time $\frac{1}{7}$-approximation algorithm.
- If the distances satisfy triangle inequality $d_{i,j} + d_{j,k} \geq d_{i,k}$, there is a polynomial-time $\frac{1}{3}$-approximation algorithm.

Knapsack

- In KNAPSACK we have a set of n items with each item i having a value v_i and a weight w_i (both positive integers) and integer W and the task is to find a subset S of items such that $\Sigma_{i \in S} w_i \leq W$ but $\Sigma_{i \in S} v_i$ is the largest possible.
- KNAPSACK has a pseudopolynomial algorithm.
- For KNAPSACK polynomial-time approximability has no limits.

Theorem. The approximation threshold for KNAPSACK is zero.

KNAPSACK

- We show that for KNAPSACK there is a polynomial-time ε-approximation algorithm for any $\varepsilon > 0$. This is based on the following pseudopolynomial algorithm.
- Let $V = \max\{v_1, \ldots, v_n\}$.
- For each $i = 0, 1, \ldots, n$ and $0 \leq v \leq nV$, define the quantity $W(i, v)$: the minimum weight attainable by selecting among the first i items so that their total value is exactly v.
- We start with $W(0, 0) = 0$ and $W(0, v) = \infty$ for all $v \neq 0$.
- Each $W(i, v)$ with $i > 0$ can be computed by
$$W(i+1, v) = \min\{W(i, v), W(i, v - v_{i+1}) + w_{i+1}\}$$
- In the end, we pick the largest v such that $W(n, v) \leq W$.
- Each entry can be computed in constant number of steps and there are $(n+1)(nV+1)$ entries. Hence, the algorithm runs in $O(n^2 V)$ time.

KNAPSACK

- The algorithm *allows trading off accuracy for speed*.
- Given an instance of KNAPSACK $x = (w_1, \ldots, w_n, W, v_1, \ldots, v_n)$ we can define the approximate instance $x' = (w_1, \ldots, w_n, W, v'_1, \ldots, v'_n)$ where the new values are $v'_i = 2^b \lfloor \frac{v_i}{2^b} \rfloor$ (the old values with their b least significant bits replaced by zeros) where b is a parameter depending on ε.
- The time required to solve x' is $\mathrm{O}(\frac{n^2 V}{2^b})$ because we can ignore the trailing zeros in v_is.
- The solution S' of x' obtained can be different from the optimal solution S of x but it can be shown that for $c(x') = \Sigma_{i \in S'} v'_i$ holds:

$$\Sigma_{i \in S} v_i \geq \Sigma_{i \in S'} v'_i \geq \Sigma_{i \in S} v_i - n2^b.$$

KNAPSACK

- Hence,

$$\frac{\mathrm{OPT}(x) - c(x')}{\mathrm{OPT}(x)} \leq \frac{\Sigma_{i \in S} v_i - (\Sigma_{i \in S} v_i - n2^b)}{\mathrm{OPT}(x)} \leq \frac{n2^b}{V}$$

  where $\mathrm{OPT}(x) \geq V$.
- So given any $\varepsilon > 0$, we truncate the last $b = \lfloor \log \frac{\varepsilon V}{n} \rfloor$ bits of the values and arrive at an ε-approximation algorithm with running time $\mathrm{O}(\frac{n^2 V}{2^b}) = \mathrm{O}(\frac{n^3}{\varepsilon})$.
- Thus, there is a polynomial-time ε-approximation algorithm for any $\varepsilon > 0$ and the approximation threshold is zero.

Approximation Schemes

Definition. A *polynomial-time approximation scheme* for an optimization problem A is an algorithm which, for each $\varepsilon > 0$ and instance x of A, returns a solution with a relative error of at most ε in time $p_\varepsilon(|x|)$ where p_ε is a polynomial depending on ε.

- In case of KNAPSACK, the time bound p_ε depends polynomially on $\frac{1}{\varepsilon}$ and the respective scheme is then called *fully polynomial*.
- For BIN PACKING, there is an approximation scheme where the time bound p_ε depends on $\frac{1}{\varepsilon}$ exponentially.

Maximum Independent Set

- INDEPENDENT SET: the approximation threshold is either zero or one.
- **Lemma.** G has an independent set of size k iff G^2 has an independent set of size k^2
  where G^2 is a graph with nodes $V \times V$ and edges
  $\{[(u,u'),(v,v')] \mid$ either $u = v$ and $[u',v'] \in E$ or $[u,v] \in E\}$
- From this it can be shown:

  Theorem. If there is an ε_0-approximation algorithm for INDEPENDENT SET for any $\varepsilon_0 < 1$, then there is a polynomial-time approximation scheme for INDEPENDENT SET.

k-DEGREE INDEPENDENT SET

- For graphs where each node has degree at most k the following algorithm works:
  Start with $I = \emptyset$.
  While there are nodes left in G, repeatedly delete from G any node v and all of its adjacent nodes adding v to I.
- The resulting I is an independent set of G.
- Since each stage adds another node to I and deletes at most $k+1$ nodes, the resulting independent set has at least $\frac{|V|}{k+1}$ nodes. This is at least $\frac{1}{k+1}$ times the true maximum independent set.
- From this it follows:

Theorem. The approximation threshold of the k-DEGREE INDEPENDENT SET problem is at most $1 - \frac{1}{k+1} = \frac{k}{k+1}$.

2. Approximation and Complexity

- A polynomial-time approximation scheme for an optimization problem is the next best thing to a polynomial-time exact algorithm for the problem.
- For **NP**-complete optimization problems an important question is whether such a scheme exists.
- We use L-reductions to order optimization problems by difficulty.

L-reductions

- An L-reduction from an optimization problem A to an optimization problem B is a pair of functions (R, S) both computable in logarithmic space satisfying the following two properties:
  (i) If x is an instance of A with optimum cost $\mathrm{OPT}(x)$, then $R(x)$ is an instance of B with optimum cost that satisfies

  $$\mathrm{OPT}(R(x)) \leq \alpha \mathrm{OPT}(x) \qquad \text{where } \alpha \text{ is a positive constant.}$$

  (ii) If s is any feasible solution of $R(x)$, then $S(s)$ is a feasible solution of x such that

  $$|\mathrm{OPT}(x) - c(S(s))| \leq \beta |\mathrm{OPT}(R(x)) - c(s)|$$

  where is β is another positive constant.
- Notice: (i) S returns a feasible solution of x which is not much more suboptimal than the given by solution of $R(x)$. (ii) If s is an optimum solution of $R(x)$, then $S(s)$ must be the optimum solution of x.

L-reductions Compose

Proposition. If (R, S) is an L-reduction from problem A to problem B and (R', S') is an L-reduction from problem B to problem C, then their composition $(R \cdot R', S \cdot S')$ is an L-reduction from A to C.

Proposition. If there is an L-reduction (R, S) from A to B with constants α and β and there is a polynomial-time ε-approximation algorithm for B, then there is a polynomial-time $\frac{\alpha\beta\varepsilon}{1-\varepsilon}$-approximation algorithm for A.

Corollary. If there is an L-reduction (R, S) from A to B and there is a polynomial-time approximation scheme for B, then there is a polynomial-time approximation scheme for A.

MAXSNP

- Fagin's theorem characterizes **NP** in terms of existential second order logic (expressions $\exists P\phi$ where ϕ is first-order).
- The *strict* fragment of **NP**, denoted **SNP**, consists of all graph-theoretic properties expressible as

$$\exists S \forall x_1 \ldots \forall x_n \, \phi(S, G, x_1, \ldots, x_n).$$

- $\mathbf{MAXSNP}_0$ is the class of optimization problems A defined by

$$\max_{S \subseteq V^r} |\{(x_1, \ldots, x_k) \in V^k \mid \phi(G_1, \ldots, G_m, S, x_1, \ldots, x_n)\}|$$

where relations $G_1, \ldots, G_m$ over finite V form the input.

Example. MAX CUT $\in \mathbf{MAXSNP}_0$ as it can be stated as

$$\max_{S \subseteq V} |\{(x,y) \in V \times V : (E(x,y) \vee E(y,x)) \wedge S(x) \wedge \neg S(y)\}|$$

where the input is V (the set of nodes) and E (the edge relation of a graph).

MAXSNP-Completeness

Theorem. Let A be a problem in $\mathbf{MAXSNP}_0$. Suppose that A is of the form $\max_S |\{(x_1, \ldots, x_n) \mid \phi\}|$. Then A has a $(1 - 2^{-k_\phi})$-approximation algorithm where k_ϕ denotes the number of atomic expressions in ϕ that involve S.

Definition. **MAXSNP** is the class of all optimization problems that are L-reducible to a problem in $\mathbf{MAXSNP}_0$.

A problem A in **MAXSNP** is **MAXSNP**-complete iff all problems in **MAXSNP** L-reduce to A.

Proposition. If a **MAXSNP**-complete problem has a polynomial-time approximation scheme, then all problems in **MAXSNP** have a polynomial-time approximation scheme.

Theorem. MAX3SAT is **MAXSNP**-complete.

It can be shown (by a non-trivial proof) that also 3-OCCURRENCE MAX3SAT is **MAXSNP**-complete.

Further MAXSNP-complete problems

Theorem. The following problems are **MAXSNP**-complete:

(a) 4-DEGREE INDEPENDENT SET

(b) 4-DEGREE NODE COVER

(c) 5-OCCURRENCE MAX2SAT

(d) MAX NAESAT

(e) MAX-CUT

3. Nonapproximability

Motivation

- Do **MAXSNP**-complete problems have polynomial-time approximation schemes?

  Answer: No, if $\mathbf{P} \neq \mathbf{NP}$.
- This (non-trivial) result is based on an alternative characterization of **NP** using *weak verifiers*.

Verifiers

- A relation R is *polynomially balanced* if $(x,y) \in R$ implies $|y| \leq |x|^k$ for some $k \geq 1$.
- Machine M is a *verifier* for L if L can be written as
$$L = \{x \mid (x,y) \in R \text{ for some } y\}$$
where R is a polynomially balanced relation decided by M.

Theorem. [*The weak verifier version of Cook's theorem.*]

A language $L \in \mathbf{NP}$ iff it has a deterministic log-space verifier.

$(\log n, 1)$-restricted verifiers

- A *$(\log n, 1)$-restricted verifier* is a randomized machine that uses
  (i) only $O(\log |x|)$ random bits and
  (ii) a constant number (k) of bits of y when verifying $(x,y) \in R$.
- Given a random bit string r ($c \log |x|$ bits), the verifier
  1. computes $Q(x,r)$, a set of k indices,
  2. chooses k symbols $y_1, \dots, y_k$ from y according to indices in $Q(x,r)$,
  3. and performs a polynomial-time computation using input x, r, and $y_1, \dots, y_k$, and answers "yes" or "no".

A new characterization of NP

Definition. A $(\log n, 1)$-restricted verifier *decides* a relation R iff for each input x and alleged certificate y,

1. $(x,y) \in R$ implies for all random strings the verifier says "yes" and
2. $(x,y) \notin R$ implies at least half of random strings make the verifier say "no".

By a very non-trivial proof it can be shown:

Theorem. A language $L \in \mathbf{NP}$ iff it has a $(\log n, 1)$-restricted verifier.

Nonapproximability Results

As a consequence of the previous theorem it can be shown:

Theorem. If there is a polynomial-time approximation scheme for MAX3SAT, then $\mathbf{P} = \mathbf{NP}$.

Some corollaries:

- If $\mathbf{P} \neq \mathbf{NP}$, then no **MAXSNP**-complete problem has a polynomial-time approximation scheme.
- Unless $\mathbf{P} = \mathbf{NP}$, the approximation threshold of INDEPENDENT SET and CLIQUE is one.

Learning Objectives

- The concept of a polynomial-time ε-approximation algorithm and approximation threshold.
- Examples of polynomial-time ε-approximation algorithms.
- The concept of an approximation scheme.
- The concepts of L-reductions and **MAXSNP**-completeness
- The concept of weak verifiers and the related nonapproximability results.