Optimization Problems and Approximation

We are unable to solve NP-complete problems efficiently, i.e., there is no known way to solve them in polynomial time. Most of them are decision versions of optimization problems... with a set of feasible solutions for each instance with an associated quality measure

Why not looking for an approximate solution? Is there a difference in complexity?

Example Knapsack revisited

All set \( T \subseteq S \) with \( \sum_{i \in T} w(i) \leq W \) are feasible solutions.

The quality of the solution \( T \) wrt. the instance \( i \) is \( \sum_{i \in T} v(i) \)

Example Problem: MaxkSat

MaxSat is NP-complete (thus MaxkSat is NP-hard):

Max3Sat(D) is a special case
But also Max2Sat(D) is NP-complete...

Optimization Problems and Approximation

Definition of Optimization Problems

\[ OPTPROB = \{ I, \text{sol,m,type} \} \]

\( I \) the instance set
\( \text{sol}(i) \) the set of feasible solutions for instance \( i \)
\( m(i,s) \) the measure of solution \( s \) wrt. instance \( i \)

\[ \text{opt}(i) = \text{type} \ m(i,s) \]

Optimization Problems and Approximation

Example Problem: MaxkSat

MaxSat is NP-complete

Max3Sat(D) is a special case
But also Max2Sat(D) is NP-complete....

Example Problem: MaxkSat

MaxSat is NP-complete

Max3Sat(D) is a special case
But also Max2Sat(D) is NP-complete....

Optimization Problems and Approximation

The Class NPO

NPO is the class of optimization problems whose decision versions are in NP.

\[ OPTPROB = \{ I, \text{sol,m,type} \} \]

\( \exists \) polynomial \( p : \forall i \in I, \exists \text{sol}(i) : |s| \leq p(|I|) \)

deciding \( s \in \text{sol}(i) \) is in \( P \)

computing \( m(s,i) \) is in \( FP \)
Example Problem: MaxkSat
NP-hardness

But also Max2Sat(D) is NP-complete....

... a local replacement reduction from 3SAT:

\((x \lor y \lor z)\) is replaced by
\((x \lor y \lor z)(x \lor y \lor z)\)

\((x \lor y \lor z)(y \lor w)(z \lor w)\)

Each 3-literal clause is replaced by a 10 clauses.

If the original clause was satisfied, then 7 in the replacement can be satisfied.

Set \(K = 7m\) where \(m\) is the number of clauses in the original.

Optimization Problems and Approximation Performance Ratio

Approximation algorithms deliver solutions of guaranteed quality – they are not heuristics.

But how to measure the quality of a solution?

Let \(O = (I, \text{sol}, m, \text{type})\) be an optimization problem. Given \(i \in I\) and a \(s \in \text{sol}(i)\) we define

\[R(i, s) = \frac{\text{opt}(i)}{\text{m}(i, s)} \frac{\text{m}(i, s)}{\text{opt}(i)}\]

as the performance ratio.

\(s \in \text{sol}(i)\) is an \(r-\)approximate solution if \(R(i, s) \leq r\).
Optimization Problems and Approximation Performance Ratio

Let \( O \) be an optimization problem given \( i \in I \) and a \( s \in \text{sol}(i) \) we define

\[
R(i, s) = \max \left( \frac{\text{opt}(i)}{\text{m}(i, s)}, \frac{\text{m}(i, s)}{\text{opt}(i)} \right)
\]

as the performance ratio. \( s \in \text{sol}(i) \) is a \( r \)-approximate solution if \( R(i, s) \leq r \).

\( R(i, s) = 1 \) implies that \( s \) is optimal.

\( R(i, s) \geq 1 \) in general, the closer to 1, the better.

Example Problem MaxkSat

MaxkSat \( \iff \) \( I, \text{sol}, m, \text{max} \) >

\( I = \text{CNF - Formulas} \) with at most \( k \) literals per clause

\( \text{sol}(\phi) = \) set of assignments to the vars. of \( \phi \)

\( m(\phi, A) = \) the number of clauses which are satisfied by \( A \)

\[
R(\phi, A) = \frac{\text{opt}(\phi)}{m(\phi, A)} \quad \text{if we have an } A \text{ with } R(\phi, A) \leq \frac{3}{2}
\]

no \( A \) can satisfy more than \( \frac{3}{2} m(\phi, A) \) clauses.

Approximation Algorithm

Example Problem: MaxkSat

approxMaxSat(\( \phi \))

1. for \( i = 1 \) to \( n \)
2. \( \text{val} := E(m(\phi, A \cup \{x_i = \text{true}\})) \rangle E(m(\phi, A \cup \{x_i = \text{false}\})) \rangle \)
3. \( A := A \cup \{ x_i = \text{val} \}; \phi := \phi \setminus x_i = \text{val} \)
4. return \( A \)

\[
E(\phi()) = \sum_{x \in \phi} 1 - 2^{-|x|} \geq \sum_{x \in \phi} 1 - 2^{-1} = \frac{1}{2} | \phi |
\]

Thus, this algorithm is a 2-approximate algorithm or better.

Approximation Algorithm

Example Problem: VertexCover

approxVertexCover(V, E)

1. \( C := \emptyset \)
2. while \( E \neq \emptyset \) do
3. \( \text{pick a } a, v \in E \) with \( a < v \)
4. \( C := C \cup \{ a, v \} \)
5. \( \text{remove } (a, v) \text{ from } V, E \)
6. return \( C \)

C is indeed a valid cover.

Every cover must cover all the edges picked in line 3.

Thus every cover must contain at least \( | C | \geq 2 \) vertexes.

\[
R(G, C) = \frac{\text{m}(G, C)}{\text{opt}(G)} \leq 2
\]

Approximation Classes

APX

We have two approximation problems, which can be solved within a constant performance ratio within polynomial time.

So it’s time to define a corresponding class: APX.

\( O \) be an \( NP \)-problem.

\( O \in APX \) if there exists an \( r \)-approximation algorithm for \( O \)

which run in polynomial time for some constant \( r \geq 1 \).
Approximation Classes

Example Problem: TSP (I)

We will show that $TSP \in APX \Leftrightarrow P = NP$.

We use another $NP$-complete problem to reduce from: $HAMILTONIANCYCLE$

$HAMILTONIANCYCLE$ : Given a graph $G = (V, E)$, is there a cycle, which visits any node exactly once?

We construct a distance matrix $M$ as follows (for $r \geq 1$):

$$M(u,v) = \begin{cases} 1; & u,v \in E \\ \lfloor r \cdot |V| \rfloor; & \text{otherwise} \end{cases}$$

Approximation Classes

Example Problem: TSP (II)

We construct a distance matrix $M$ as follows (for $r \geq 1$):

$$M(u,v) = \begin{cases} 1; & u,v \in E \\ \lfloor r \cdot |V| \rfloor; & \text{otherwise} \end{cases}$$

If $G$ is a positive instance, then $\text{opt}(M) = |V| \\
Otherwise $\text{opt}(M) \geq \lceil r \cdot |V| \rceil - 1$.

Now assume that there is an $r$-approximate algorithm for $TSP$.

Approximation Classes

Example Problem: TSP (III)

If $G$ is a positive instance, then $\text{opt}(M) = |V| \\
Otherwise $\text{opt}(M) \geq \lceil r \cdot |V| \rceil - 1$.

Now assume that there is an $r$-approximate algorithm approx for $TSP$ and let $s = \text{approx}(M)$.

If $G \in HAMILTONIANCYCLE$, we find

$$r \geq \frac{m(M,s)}{\text{opt}(M)} - \frac{m(M,s)}{|V|} \quad \text{and so} \quad |V| \cdot r \geq m(M,s).$$

But otherwise we have

$$m(M,s) \geq \text{opt}(M) \geq \lceil r \cdot |V| \rceil - 1 > \lfloor r \cdot |V| \rfloor.$$

Approximation Classes

Example Problem: TSP (IV)

So we could prove that $TSP \in APX$ (assuming $P \neq NP$) by giving a reduction from an $NP$-hard problem, which established a gap between positive and negative instances.

The gap was large enough to distinguish whether we reduced from a positive or a negative instance.

Wanted: A generic reduction from $NP$-hard problems, to approximation problems which produces gaps.

Approximation Classes

Relationships

$$APX \subseteq NP$$

$$TSP \in APX \Leftrightarrow P = NP$$

$$APX \subseteq NP \Leftrightarrow P \neq NP$$

Max3Sat and VertexCover are in $APX$.

Approximation Classes

Approximation Schemes

An algorithm which can be parametrized with the performance ratio to achieve is called approximation scheme.

Let $O$ be an optimization problem. Then an algorithm $A$ is an approximation scheme for $O$ if for all $i \in I$, $r > 1$ and $s = A(i,r)$

$$s \in \text{sol}(i) \text{ and } R(i,s) \leq r.$$

Max3Sat and VertexCover are in $APX$. 

$4$
Approximation Schemes
The classes PTAS and FPTAS

\[ O \in \text{PTAS} \text{ if there is an approximation scheme } A \text{ such that } A(i,n) \text{ runs in } \text{DTIME}(\text{poly}(n)) \text{ for all } i \leq n \text{ and } n > 1. \]

\[ O \in \text{PTAS} \text{ if there is an approximation scheme } A \text{ such that } A(i,r) \text{ runs in } \text{DTIME}(\text{poly}(r)) \text{ for all } i \leq n \text{ and any fixed } r > 1. \]

Example Problem: KNAPSACK
A Pseudo-Polynomial Algorithm

Let \( W(i, v) \) be the minimum weight attainable by selecting among the first \( i \) items such that their total value is exactly \( v \).

\[ W(0,0) = 0 \]
\[ W(0,v) = v \leq 0 \]
\[ W(i+1, v) = \min[W(i,v), W(i, v - v(i+1)) + w(i+1)] \]

By building the table of the \( W(i, v) \) for \( 0 \leq i \leq n \) and \( 0 \leq v \leq V = \sum_{i=1}^{n} v(i) \) we can solve \( \text{KNAPSACK} \).

This algorithm runs in \( \text{DTIME}(\text{poly}(n, V)) \). (pseudo - poly.)

Example Problem: KNAPSACK
An FPTAS (I)

This algorithm runs in \( \text{DTIME}(\text{poly}(n, V)) \). (pseudo - poly.)

Assume \( \epsilon > 0 \) fixed.

Let \( l = \left\lceil \log \max_{i} v(i) \right\rceil \).

Choose \( k \) with \( \frac{n}{2^k} < \epsilon \).

Set \( L = l - k \log n \).

Define \( \nu(i) = \left\lfloor \frac{v(i)}{2^k} \right\rfloor \).

Solving \( I' \) optimally yields an \( 1 + \epsilon \) approximate solution for \( I \).

Example Problem: KNAPSACK
An FPTAS (II)

This algorithm runs in \( \text{DTIME}(\text{poly}(n, V)) \). (pseudo - poly.)

Assume \( \epsilon > 0 \) fixed.

Let \( l = \left\lceil \log \max_{i} v(i) \right\rceil \).

Choose \( k \) with \( \frac{n}{2^k} < \epsilon \).

Set \( L = l - k \log n \).

Define \( \nu(i) = \left\lfloor \frac{v(i)}{2^k} \right\rfloor \).

Solving \( I' \) optimally yields an \( 1 + \epsilon \) approximate solution for \( I \).

Example Problem: KNAPSACK
An FPTAS (III)

This algorithm runs in \( \text{DTIME}(\text{poly}(n, V)) \). (pseudo - poly.)

Assume \( \epsilon > 0 \) fixed.

Let \( l = \left\lceil \log \max_{i} v(i) \right\rceil \).

Choose \( k \) with \( \frac{n}{2^k} < \epsilon \).

Set \( L = l - k \log n \).

Define \( \nu(i) = \left\lfloor \frac{v(i)}{2^k} \right\rfloor \).

Solving \( I' \) optimally yields an \( 1 + \epsilon \) approximate solution for \( I \) within \( \text{DTIME}(\text{poly}(l, 1/\epsilon)) \). \( \text{KNAPSACK} \in \text{PTAS} \).
Approximation Schemes

Polynomially Bound Problems

Let $O$ be a problem in $\text{NPO}$. If there is polynomial $p$ such that
$\forall i \in I, x \in \text{sol}(i), m(i, x) \leq p(|i|)$
then $O$ is polynomially bound, i.e.,
$O \in \text{NPO} - PB$.

If there is an $NP - hard$ problem in $\text{NPO} - PB$
which admits an FPTAS, then $P = NP$.

Polynomially Bound Problems

Permit no FPTAS (I)

If there is an $NP - hard$ problem in $\text{NPO} - PB$
which admits an FPTAS, then $P = NP$.

Let $O$ be a maximization problem in $\text{NPO} - PB$.
Set $r(i) = 1 + \frac{1}{p(|i|)}$ where $p$ is the poly. - bound.

An $r(i)$-approximate solution $x$ for $i$ is optimal since,
$p(|i|) = r(i) \geq \frac{\text{opt}(i)}{m(i)}$ gives
$m(i) \geq \text{opt}(i) - \frac{\text{opt}(i)}{p(|i|) + 1} > \text{opt}(i) - 1$

If $O$ would be in FPTAS then we can solve $O$ optimally
in $\text{DTIME}(\text{poly}(|i|)) + 1 = \text{DTIME}(\text{poly}(|i|))$.

Polynomially Bound Problems

Permit no FPTAS (II)

Set $r(i) = 1 + \frac{1}{p(|i|)}$ where $p$ is the poly. - bound.

An $r(i)$-approximate solution $x$ for $i$ is optimal since,
$p(|i|) + 1 = r(i) \geq \frac{\text{opt}(i)}{m(i)}$
$m(i, x) \geq \text{opt}(i) - \frac{\text{opt}(i)}{p(|i|) + 1} > \text{opt}(i) - 1$

Approximation Classes

Problems in PTAS-FPTAS

PLANAR INDEPENDENTSET is in $\text{NPO} - PB$ and is $NP - hard$.
PLANAR INDEPENDENTSET $\in$ FPTAS $\Rightarrow P = NP$.

Unproven : PLANAR INDEPENDENTSET $\in$ PTAS.

Approximation Classes

Relationships

$\text{FPTAS} \subseteq \text{PTAS} \subseteq \text{APX} \subseteq \text{NPO}$

TSP $\in$ APX $\Leftrightarrow P = NP$
Max3Sat $\in$ FPTAS $\Leftrightarrow P = NP$

Two questions: Are there problems in PTAS-FPTAS?
Are there problems in APX - PTAS?
(as usual, based on $P \neq NP$)
Hardness in Approximation

Wanted: A generic reduction from \( NP \) – hard problems, to approximation problems which produces gaps.

Remember the reduction to TSP...

Relies on the so-called PCP-Theorem – an alternative formulation of NP.

It allows to reduce \( NP \) – complete languages to approximation problems.

---

Hardness in Approximation

PCP-Theorem (I)

A language \( L \) is in \( \text{PCP}(r(n), q(n)) \) if there is a polynomial time \( \text{PCP}(r(n), q(n)) \)-Verifier \( V \) such that

\[
\begin{align*}
\forall x \in L. \ & \exists \Pi : P_q[V(x, \Pi, \overline{R}) = \text{accept}] = 1 \\
\forall x \notin L. & \forall \Pi : P_q[V(x, \Pi, \overline{R}) = \text{accept}] \leq 1/2 \\
\text{with } \overline{R} = O(r |x|), \text{ and } V \text{ reading } O(q(n)) \text{ bits non-adaptively from } \Pi.
\end{align*}
\]

Easy: \( NP \subseteq \text{PCP}(\log n, 1) \)

Hard: \( NP \not\subseteq \text{PCP}(\log n, 1) \)

---

Hardness in Approximation

PCP-Theorem (II)

A language \( L \) is in \( \text{PCP}(r(n), q(n)) \) if there is a polynomial time \( \text{PCP}(r(n), q(n)) \)-Verifier \( V \) such that

\[
\begin{align*}
\forall x \in L. \ & \exists \Pi : P_q[V(x, \Pi, \overline{R}) = \text{accept}] = 1 \\
\forall x \notin L. & \forall \Pi : P_q[V(x, \Pi, \overline{R}) = \text{accept}] \leq 1/2 \\
\text{with } \overline{R} = O(r |x|), \text{ and } V \text{ reading } O(q(n)) \text{ bits non-adaptively from } \Pi.
\end{align*}
\]

- \( \text{PCP - Theorem: } NP \subseteq \text{PCP}(\log n, 1) \)

---

Hardness in Approximation

Example Problem: Max3Sat (I)

Observe that the once the \( O(q(n)) \) bits have been read from the proof \( \Pi \), the decision of \( V \) is only depending on them.

Thus we can define a set of Boolean Expressions

\[
\phi(s, \overline{R} | \overline{\overline{P}}) \text{ where}
\begin{align*}
s \text{ is the input}, \\
\overline{R} \text{ is the random string of length } O(\log n), \\
\overline{\overline{P}} \text{ are the bits read in } \Pi, \\
\phi(s, \overline{R} | \overline{\overline{P}}) = 1 \iff V(s, \Pi, \overline{R}) = \text{accept}.
\end{align*}
\]
Example Problem: Max3Sat (II)

Hardness in Approximation

Each \( \phi(x, \overline{R}) \) can be expressed by \( d \) clauses, where \( d \) is constant (since \( |\overline{P}| \) is constant).

Let \( \phi \) be the conjunction of the expressions \( \phi(x, \overline{R}) \) for all \( \overline{R} \) (\( \overline{R} \leftarrow \epsilon \log n \)).

\[ x \in L \Rightarrow \exists \overline{P}: P_{\phi}[v(x, \Pi, \overline{R})] = \text{accept} \leq 1 \]

\[ \Rightarrow \forall \phi(x, \overline{R}) \text{ can be satisfied simultaneously} \]

\[ \Rightarrow \phi \text{ satisfiable.} \]

Example Problem: Max3Sat (III)

Hardness in Approximation

Each \( \phi(x, \overline{R}) \) can be expressed by \( d \) clauses, where \( d \) is constant (since \( |\overline{P}| \) is constant).

Let \( \phi \) be the conjunction of the expressions \( \phi(x, \overline{R}) \) for all \( \overline{R} \) (\( \overline{R} \leftarrow \epsilon \log n \)).

\[ x \notin L \Rightarrow \forall \overline{P}: P_{\phi}[v(x, \Pi, \overline{R})] = \text{accept} \leq 1/2 \]

\[ \Rightarrow \forall \phi(x, \overline{R}) \text{ unsatisfied.} \]

\[ \Rightarrow \exists \phi(x, \overline{R}) \text{ unsatisfied.} \]

Example Problem: Max3Sat (IV)

Hardness in Approximation

Each \( \phi(x, \overline{R}) \) can be expressed by \( d \) clauses, where \( d \) is constant (since \( |\overline{P}| \) is constant).

Let \( \phi \) be the conjunction of the expressions \( \phi(x, \overline{R}) \) for all \( \overline{R} \) (\( \overline{R} \leftarrow \epsilon \log n \)).

\[ x \in L \Rightarrow \exists \overline{P}: P_{\phi}[v(x, \Pi, \overline{R})] = \text{accept} \leq 1 \]

\[ \Rightarrow \forall \phi(x, \overline{R}) \text{ can be satisfied simultaneously} \]

\[ \Rightarrow \phi \text{ satisfiable.} \]

Remark: Decoding of PCP-Proofs

Given a proof \( \Pi \) with \( P_{\phi}[v(x, \overline{R})] = \text{accept} \leq 1/2 \)

\[ \Rightarrow \exists \phi(x, \overline{R}) \text{ unsatisfied.} \]

\[ \Rightarrow \exists \phi(x, \overline{R}) \text{ unsatisfied.} \]

Approximation Classes Relationships

\[ \text{PP} \subseteq \text{PTAS} \subseteq \text{APX} \subseteq \text{NPO} \]

\[ \text{TSP} \in \text{APX} \Leftrightarrow P = \text{NP} \]

\[ \text{Max3Sat} \in \text{PTAS} \Leftrightarrow P = \text{NP} \]

\[ \text{PLANAR INDEPSET} \in \text{PTAS} \]

\[ \Rightarrow P = \text{NP} \]
Approximation Classes
More Classes

Let $O$ be an $NPO$ problem. $O \in F \subseteq APX$ iff there exists an $f$–approximation algorithm for $O$ which run in polynomial time for some function $f \in F$.

$\text{FPTAS} \subseteq \text{PTAS} \subseteq \text{APX} \subseteq \log \text{– APX} \subseteq \text{poly} \text{– APX} \subseteq \text{exp} \text{– APX} \subseteq \text{NPO}$

Approximation Classes
More Example Problems

$\text{PLANAR INDEPENDENT SET} \quad \text{SET COVER} \quad \text{TSP}$

$\text{FPTAS} \subseteq \text{PTAS} \subseteq \log \text{– APX} \subseteq \text{poly} \text{– APX} \subseteq \text{exp} \text{– APX} \subseteq \text{NPO}$

$\text{KNAP SACK} \quad \text{MAX SAT} \quad \text{COLORING} \quad \text{MAX ONES SAT}$

Approximation Classes
More Relationships

$\text{FPTAS} \subseteq \text{PTAS} \subseteq \text{APX} \subseteq \log \text{– APX} \subseteq \text{poly} \text{– APX} \subseteq \text{exp} \text{– APX} \subseteq \text{NPO}$

$\text{FPTAS} \subseteq \text{PTAS} \subseteq \text{APX} \subseteq \log \text{– APX} \subseteq \text{poly} \text{– APX} \subseteq \text{exp} \text{– APX} \subseteq \text{NPO}$

iff

$P \neq \text{NP}$