NP-completeness

Abhiram Ranade

April 5, 2016

Where we are

Where we are

Goal: Understand problems such as IS, VC, ILP, SAT, ...

Where we are

Goal: Understand problems such as IS, VC, ILP, SAT, ...

- All these problems are reducible to each other.

Where we are

Goal: Understand problems such as IS, VC, ILP, SAT, ...

- All these problems are reducible to each other.

Many clever people have worked on each problem and by reduction on all problems.

Where we are

Goal: Understand problems such as IS, VC, ILP, SAT, ...

- All these problems are reducible to each other.

Many clever people have worked on each problem and by reduction on all problems.

So probably no polytime algorithm exists for them.

Where we are

Goal: Understand problems such as IS, VC, ILP, SAT, ...

- All these problems are reducible to each other.

  Many clever people have worked on each problem and by reduction on all problems.

  So probably no polytime algorithm exists for them.

- They are puzzle-like, and are in class NP.

Where we are

Goal: Understand problems such as IS, VC, ILP, SAT, ...

- All these problems are reducible to each other.

  Many clever people have worked on each problem and by reduction on all problems.

  So probably no polytime algorithm exists for them.

- They are puzzle-like, and are in class NP.

  NP: Class of problems having polytime verifiers.

Where we are

Goal: Understand problems such as IS, VC, ILP, SAT, ...

- All these problems are reducible to each other.

  Many clever people have worked on each problem and by reduction on all problems.

  So probably no polytime algorithm exists for them.
- They are puzzle-like, and are in class NP.

  NP: Class of problems having polytime verifiers.

Where we are

Goal: Understand problems such as IS, VC, ILP, SAT, ...

- All these problems are reducible to each other.

  Many clever people have worked on each problem and by reduction on all problems.

  So probably no polytime algorithm exists for them.

- They are puzzle-like, and are in class NP.

  NP: Class of problems having polytime verifiers.

- Cook-Levin Theorem: If $Q \in NP$ then $Q \leq_K CSAT$.

Where we are

Goal: Understand problems such as IS, VC, ILP, SAT, ...

- All these problems are reducible to each other.

  Many clever people have worked on each problem and by reduction on all problems.

  So probably no polytime algorithm exists for them.
- They are puzzle-like, and are in class NP.
  NP: Class of problems having polytime verifiers.
- Cook-Levin Theorem: If $Q \in NP$ then $Q \leq_K CSAT$.

Implications of Cook Levin:

Where we are

Goal: Understand problems such as IS, VC, ILP, SAT, ...

- All these problems are reducible to each other.

  Many clever people have worked on each problem and by reduction on all problems.

  So probably no polytime algorithm exists for them.

- They are puzzle-like, and are in class NP.

  NP: Class of problems having polytime verifiers.

- Cook-Levin Theorem: If $Q \in NP$ then $Q \leq_K CSAT$.

Implications of Cook Levin:

- If we want to reduce some problem to CSAT, we just need to show that it is in NP.

Where we are

Goal: Understand problems such as IS, VC, ILP, SAT, ...

- All these problems are reducible to each other.

  Many clever people have worked on each problem and by reduction on all problems.

  So probably no polytime algorithm exists for them.

- They are puzzle-like, and are in class NP.
  NP: Class of problems having polytime verifiers.
- Cook-Levin Theorem: If $Q \in NP$ then $Q \leq_K CSAT$.

Implications of Cook Levin:

- If we want to reduce some problem to CSAT, we just need to show that it is in NP.
- CSAT is a hardest problem in NP: if we find polytime algorithm for CSAT, we get polytime algorithms for all of NP.

Where we are

Goal: Understand problems such as IS, VC, ILP, SAT, ...

- All these problems are reducible to each other.

  Many clever people have worked on each problem and by reduction on all problems.

  So probably no polytime algorithm exists for them.

- They are puzzle-like, and are in class NP.
  NP: Class of problems having polytime verifiers.
- Cook-Levin Theorem: If $Q \in NP$ then $Q \leq_K CSAT$.

Implications of Cook Levin:

- If we want to reduce some problem to CSAT, we just need to show that it is in NP.
- CSAT is a hardest problem in NP: if we find polytime algorithm for CSAT, we get polytime algorithms for all of NP.

  NP does contain "easy" problems also, e.g. class P.

NP-hard Problems

NP-hard Problems

Definition: A problem Q is said to be NP-hard if every problem R $\in$ NP reduces to Q.

NP-hard Problems

Definition: A problem Q is said to be NP-hard if every problem R $\in$ NP reduces to Q.

Theorem: CSAT is NP-hard.

NP-hard Problems

Definition: A problem Q is said to be NP-hard if every problem R $\in$ NP reduces to Q.

Theorem: CSAT is NP-hard.
Proof:

NP-hard Problems

Definition: A problem Q is said to be NP-hard if every problem R $\in$ NP reduces to Q.

Theorem: CSAT is NP-hard.
Proof: By Cook-Levin, all R $\in$ NP reduce to CSAT. $\square$

NP-hard Problems

Definition: A problem Q is said to be NP-hard if every problem R $\in$ NP reduces to Q.

Theorem: CSAT is NP-hard.
Proof: By Cook-Levin, all R $\in$ NP reduce to CSAT. $\square$

Theorem: IS is NP-hard

NP-hard Problems

Definition: A problem Q is said to be NP-hard if every problem R $\in$ NP reduces to Q.

Theorem: CSAT is NP-hard.
Proof: By Cook-Levin, all R $\in$ NP reduce to CSAT. $\square$

Theorem: IS is NP-hard
Proof: We showed that CSAT $\leq_K$ IS.

NP-hard Problems

Definition: A problem Q is said to be NP-hard if every problem R $\in$ NP reduces to Q.

Theorem: CSAT is NP-hard.
Proof: By Cook-Levin, all R $\in$ NP reduce to CSAT. $\square$

Theorem: IS is NP-hard
Proof: We showed that CSAT $\leq_K$ IS.
But all R $\in$ NP reduce to CSAT.

NP-hard Problems

Definition: A problem Q is said to be NP-hard if every problem R $\in$ NP reduces to Q.

Theorem: CSAT is NP-hard.
Proof: By Cook-Levin, all R $\in$ NP reduce to CSAT. $\square$

Theorem: IS is NP-hard
Proof: We showed that CSAT $\leq_K$ IS.
But all R $\in$ NP reduce to CSAT.
Thus all R $\in$ NP reduce to IS by transitivity. $\square$

NP-hard Problems

Definition: A problem Q is said to be NP-hard if every problem R $\in$ NP reduces to Q.

Theorem: CSAT is NP-hard.
Proof: By Cook-Levin, all R $\in$ NP reduce to CSAT. $\square$

Theorem: IS is NP-hard
Proof: We showed that CSAT $\leq_K$ IS.
But all R $\in$ NP reduce to CSAT.
Thus all R $\in$ NP reduce to IS by transitivity. $\square$

Note:

NP-hard Problems

Definition: A problem Q is said to be NP-hard if every problem R $\in$ NP reduces to Q.

Theorem: CSAT is NP-hard.
Proof: By Cook-Levin, all R $\in$ NP reduce to CSAT. $\square$

Theorem: IS is NP-hard
Proof: We showed that CSAT $\leq_K$ IS.
But all R $\in$ NP reduce to CSAT.
Thus all R $\in$ NP reduce to IS by transitivity. $\square$

Note:

- If you want to say, "probably there is no polytime algorithm for problem Q", show that Q is NP-hard.

NP-hard Problems

Definition: A problem Q is said to be NP-hard if every problem R $\in$ NP reduces to Q.

Theorem: CSAT is NP-hard.
Proof: By Cook-Levin, all R $\in$ NP reduce to CSAT. $\square$

Theorem: IS is NP-hard
Proof: We showed that CSAT $\leq_K$ IS.
But all R $\in$ NP reduce to CSAT.
Thus all R $\in$ NP reduce to IS by transitivity. $\square$

Note:

- If you want to say, "probably there is no polytime algorithm for problem Q", show that Q is NP-hard.
- Strategy for proving that a problem R is NP-Hard:

NP-hard Problems

Definition: A problem Q is said to be NP-hard if every problem R $\in$ NP reduces to Q.

Theorem: CSAT is NP-hard.
Proof: By Cook-Levin, all R $\in$ NP reduce to CSAT. $\square$

Theorem: IS is NP-hard
Proof: We showed that CSAT $\leq_K$ IS.
But all R $\in$ NP reduce to CSAT.
Thus all R $\in$ NP reduce to IS by transitivity. $\square$

Note:

- If you want to say, "probably there is no polytime algorithm for problem Q", show that Q is NP-hard.
- Strategy for proving that a problem R is NP-Hard:
  Prove that some NPC problem reduces to R.

NP-completeness

NP-completeness

Definition: A problem Q is said to be NP-complete if it is in NP, and is NP-hard.

NP-completeness

Definition: A problem Q is said to be NP-complete if it is in NP, and is NP-hard.

"X-complete" = "containing the essence of X"

NP-completeness

Definition: A problem Q is said to be NP-complete if it is in NP, and is NP-hard.

"X-complete" = "containing the essence of X"

Theorem: CSAT is NP-complete.

NP-completeness

Definition: A problem Q is said to be NP-complete if it is in NP, and is NP-hard.

"X-complete" = "containing the essence of X"

Theorem: CSAT is NP-complete.
Proof: We showed that CSAT is NP-hard and $\in$ NP. $\square$

NP-completeness

Definition: A problem Q is said to be NP-complete if it is in NP, and is NP-hard.

"X-complete" = "containing the essence of X"

Theorem: CSAT is NP-complete.
Proof: We showed that CSAT is NP-hard and $\in$ NP. $\square$

Theorem: IS is NP-complete

NP-completeness

Definition: A problem Q is said to be NP-complete if it is in NP, and is NP-hard.

"X-complete" = "containing the essence of X"

Theorem: CSAT is NP-complete.
Proof: We showed that CSAT is NP-hard and $\in$ NP. $\square$

Theorem: IS is NP-complete
Proof: We showed that IS is NP-hard and $\in$ NP. $\square$

NP-completeness

Definition: A problem Q is said to be NP-complete if it is in NP, and is NP-hard.

"X-complete" = "containing the essence of X"

Theorem: CSAT is NP-complete.
Proof: We showed that CSAT is NP-hard and $\in$ NP. $\square$

Theorem: IS is NP-complete
Proof: We showed that IS is NP-hard and $\in$ NP. $\square$

Note:

NP-completeness

Definition: A problem Q is said to be NP-complete if it is in NP, and is NP-hard.

"X-complete" = "containing the essence of X"

Theorem: CSAT is NP-complete.
Proof: We showed that CSAT is NP-hard and $\in$ NP. $\square$

Theorem: IS is NP-complete
Proof: We showed that IS is NP-hard and $\in$ NP. $\square$

Note:

- Strategy for proving a problem NP-complete:

NP-completeness

Definition: A problem Q is said to be NP-complete if it is in NP, and is NP-hard.

"X-complete" = "containing the essence of X"

Theorem: CSAT is NP-complete.
Proof: We showed that CSAT is NP-hard and $\in$ NP. $\square$

Theorem: IS is NP-complete
Proof: We showed that IS is NP-hard and $\in$ NP. $\square$

Note:

- Strategy for proving a problem NP-complete:
  Show that it is NP-hard, and in NP.

NP-completeness

Definition: A problem Q is said to be NP-complete if it is in NP, and is NP-hard.

"X-complete" = "containing the essence of X"

Theorem: CSAT is NP-complete.
Proof: We showed that CSAT is NP-hard and $\in$ NP. $\square$

Theorem: IS is NP-complete
Proof: We showed that IS is NP-hard and $\in$ NP. $\square$

Note:

- Strategy for proving a problem NP-complete:
  Show that it is NP-hard, and in NP.
  Alternatively: Show it is in NP, and reducible from an NPC problem.

NP-completeness

Definition: A problem Q is said to be NP-complete if it is in NP, and is NP-hard.

"X-complete" = "containing the essence of X"

Theorem: CSAT is NP-complete.
Proof: We showed that CSAT is NP-hard and $\in$ NP. $\square$

Theorem: IS is NP-complete
Proof: We showed that IS is NP-hard and $\in$ NP. $\square$

Note:

- Strategy for proving a problem NP-complete:
  Show that it is NP-hard, and in NP.
  Alternatively: Show it is in NP, and reducible from an NPC problem.
- If you prove a problem NP-hard, you have shown that it is as hard to design polytime algorithms for the problem as any problem in NP, and no harder.