Path Detection: A Quantum Computing Primitive

Shelby Kimmel

University of Maryland

Joint Center for Quantum Information and Computer Science

LFQIS
06/19/2017

Things Quantum Computers are Good at:

- Factoring
  - Exponential speed-up over known classical algorithms
  - Can be used to break most commonly used public key crypto systems
- Simulating chemistry
  - Exponential speed-up over known classical algorithms
  - Useful for drug development, better carbon sequestration

“How will a quantum computer help me do X?”

“How will a quantum computer help me do X?”

- Need quantum algorithmic primitives

“How will a quantum computer help me do X?”

- Need quantum algorithmic primitives
  1. Apply to a wide range of problems
  2. Easy to understand and analyze (without knowing quantum mechanics)

"How will a quantum computer help me do X?"

- Need quantum algorithmic primitives
    1. Apply to a wide range of problems
    2. Easy to understand and analyze (without knowing quantum mechanics)

  – Ex: Searching unordered list of n items
    – Classically, takes $O(n)$ time
    – Quantumly, takes $O(\sqrt{n})$ time

“How will a quantum computer help me do X?”

- Need quantum algorithmic primitives
  1. Apply to a wide range of problems
  2. Easy to understand and analyze (without knowing quantum mechanics)

  – Ex: Searching unordered list of n items
    – Classically, takes $O(n)$ time
    – Quantumly, takes $O(\sqrt{n})$ time

- New primitive: ***st*-connectivity**

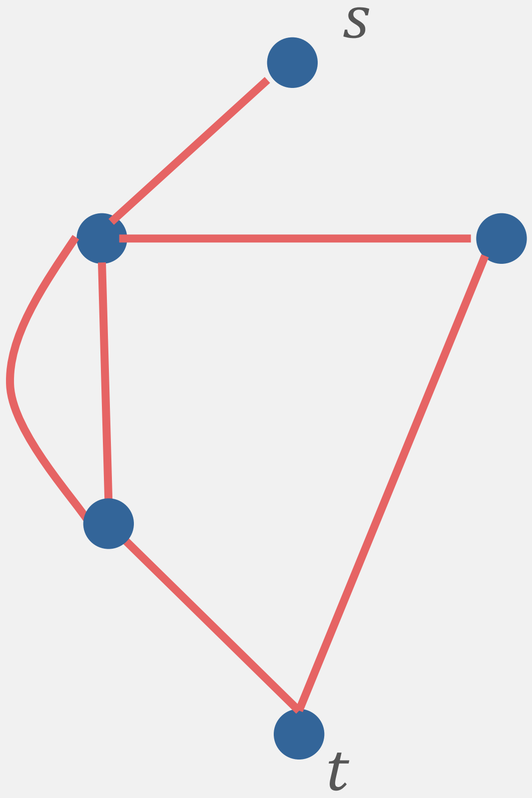
Outline:

A. Introduction to st-connectivity
B. st-connectivity makes a good algorithmic primitive
   1. Applies to a wide range of problems

   2. Easy to understand (without knowing quantum mechanics)

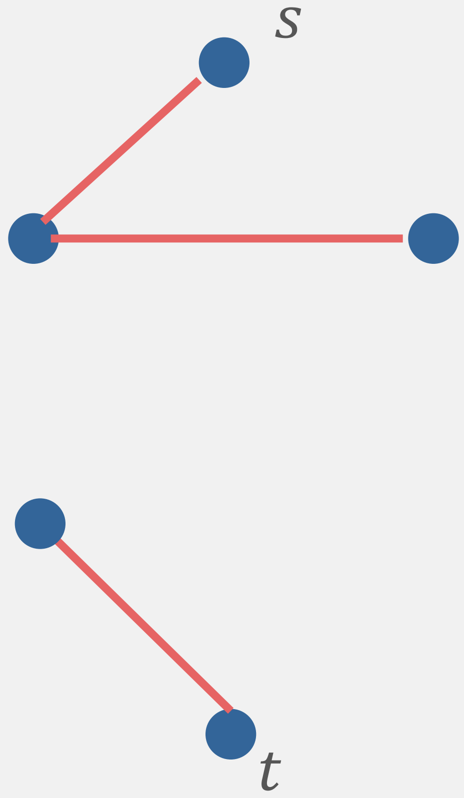
st-connectivity

$\boldsymbol{st-connectivity}$:
is there a path from s to t?



st-connectivity

$\boldsymbol{st-connectivity}$:
is there a path from s to t?

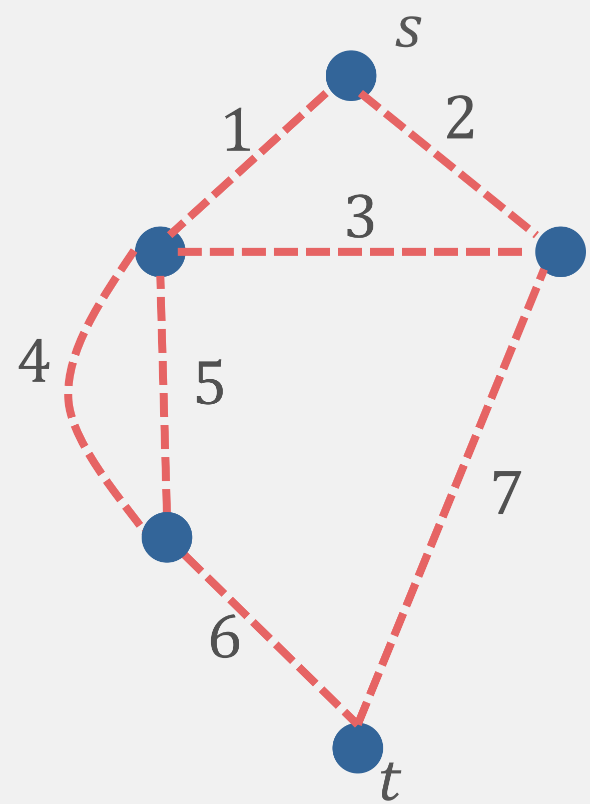


Black Box Model

$$i \rightarrow \boxed{G} \rightarrow e_i$$

Edge label

- $e_i = 1$ if i^{th} edge is there
- $e_i = 0$ if edge is not there



Let $\mathcal{H}$ be the set of graphs G that the black box might contain.

Figure of Merit

- Query Complexity
  - Number of uses (queries) of the black box
  - All other operations are free
  - Always a lower bound on time complexity (situation when other operations are not free)
  - Often (but not always) a good proxy for time complexity

- Under mild assumption, for our algorithm,

  quantum query complexity ≅ quantum time complexity

- In query model it is easier to *prove*
  - Quantum-to-classical speed-ups
  - Optimality

Outline:

A. Introduction to st-connectivity
B. st-connectivity makes a good algorithmic primitive
    1. Applies to a wide range of problems

    2. Easy to understand (without knowing quantum mechanics)

Outline:

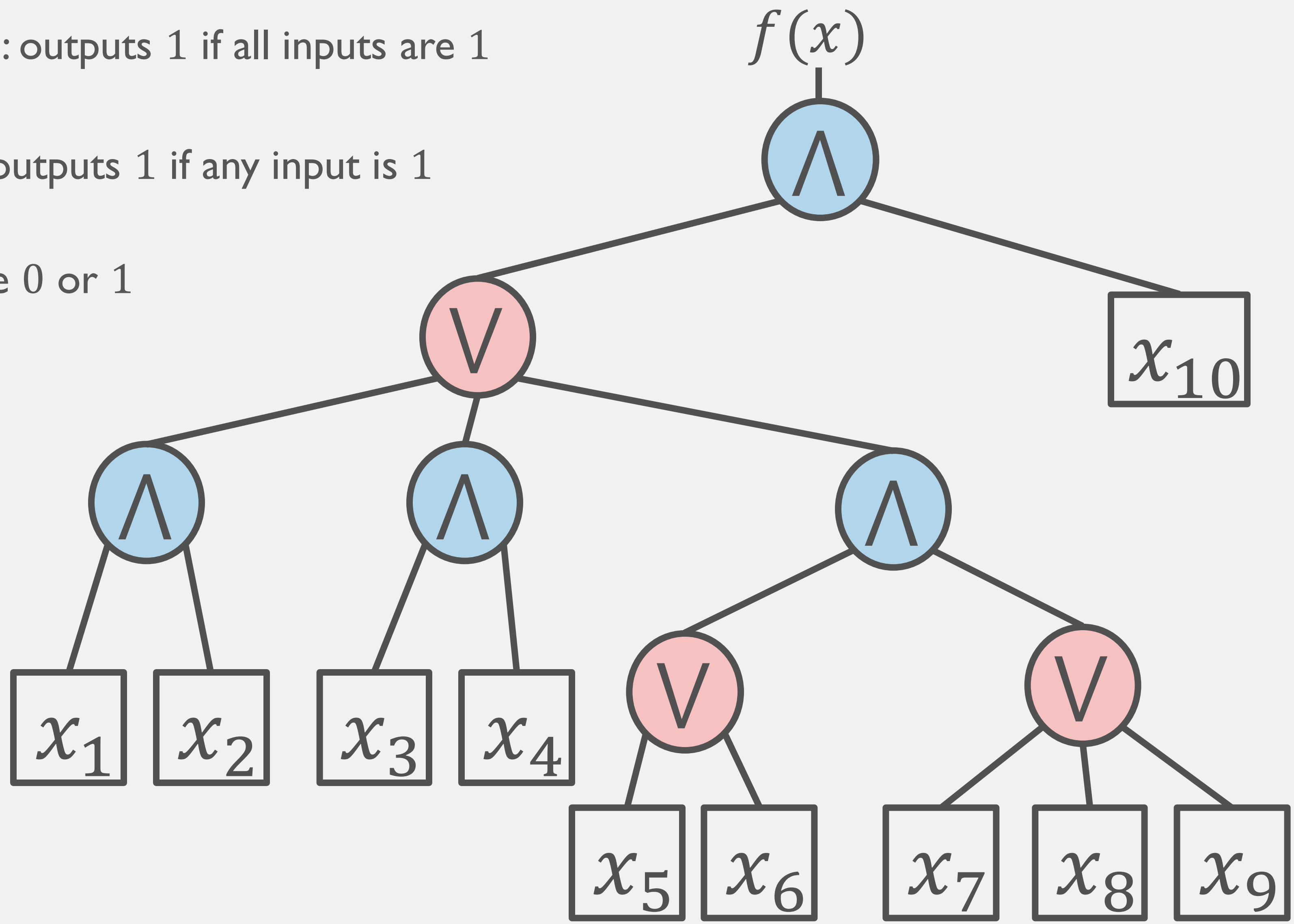
A. Introduction to st-connectivity
B. st-connectivity makes a good algorithmic primitive
   1. Applies to a wide range of problems
      - Evaluating Boolean formulas reduces to st-connectivity
   2. Easy to understand (without knowing quantum mechanics)

Boolean Formulas

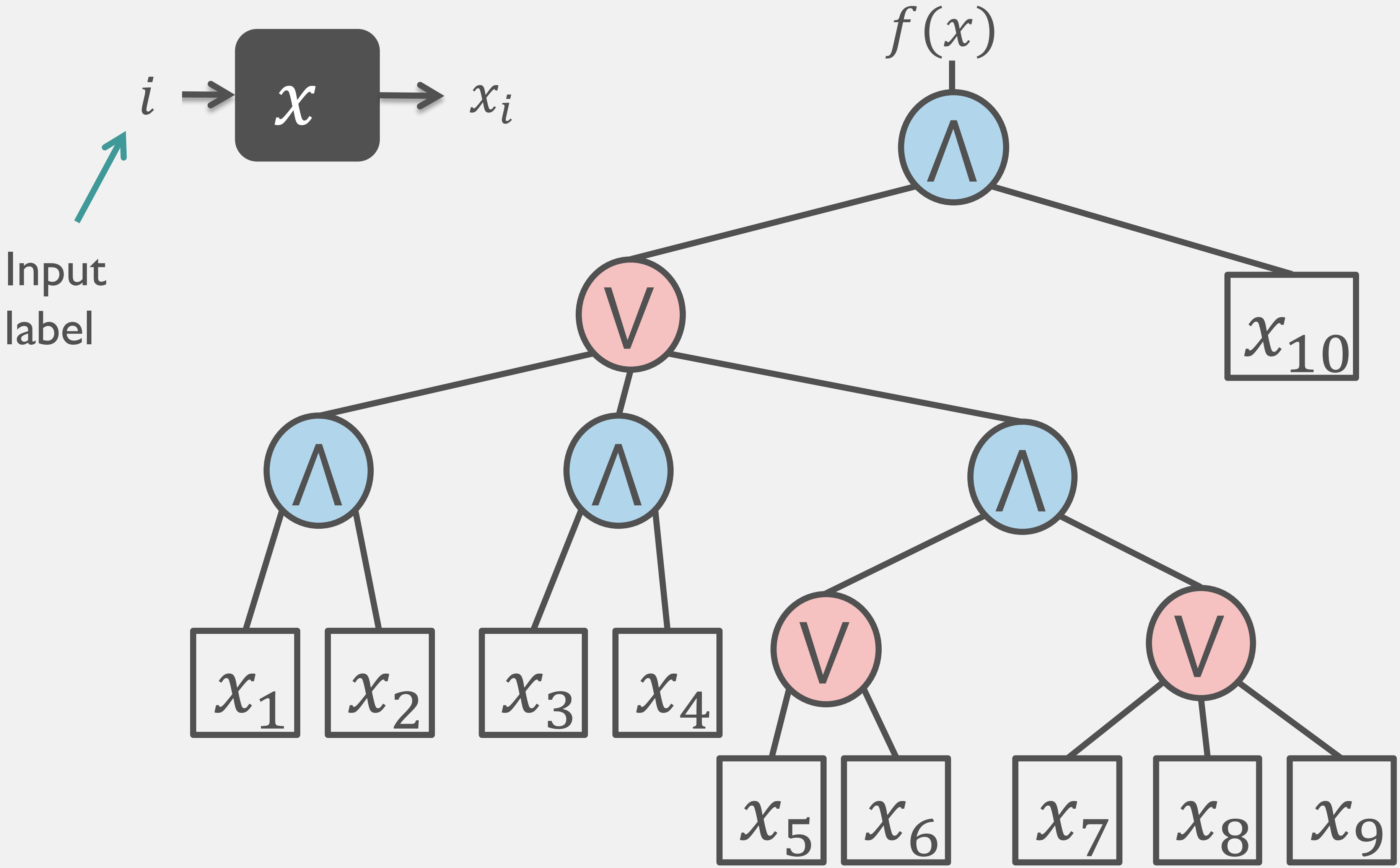
$\wedge$ AND: outputs 1 if all inputs are 1

$\vee$ OR: outputs 1 if any input is 1

x_i Value 0 or 1

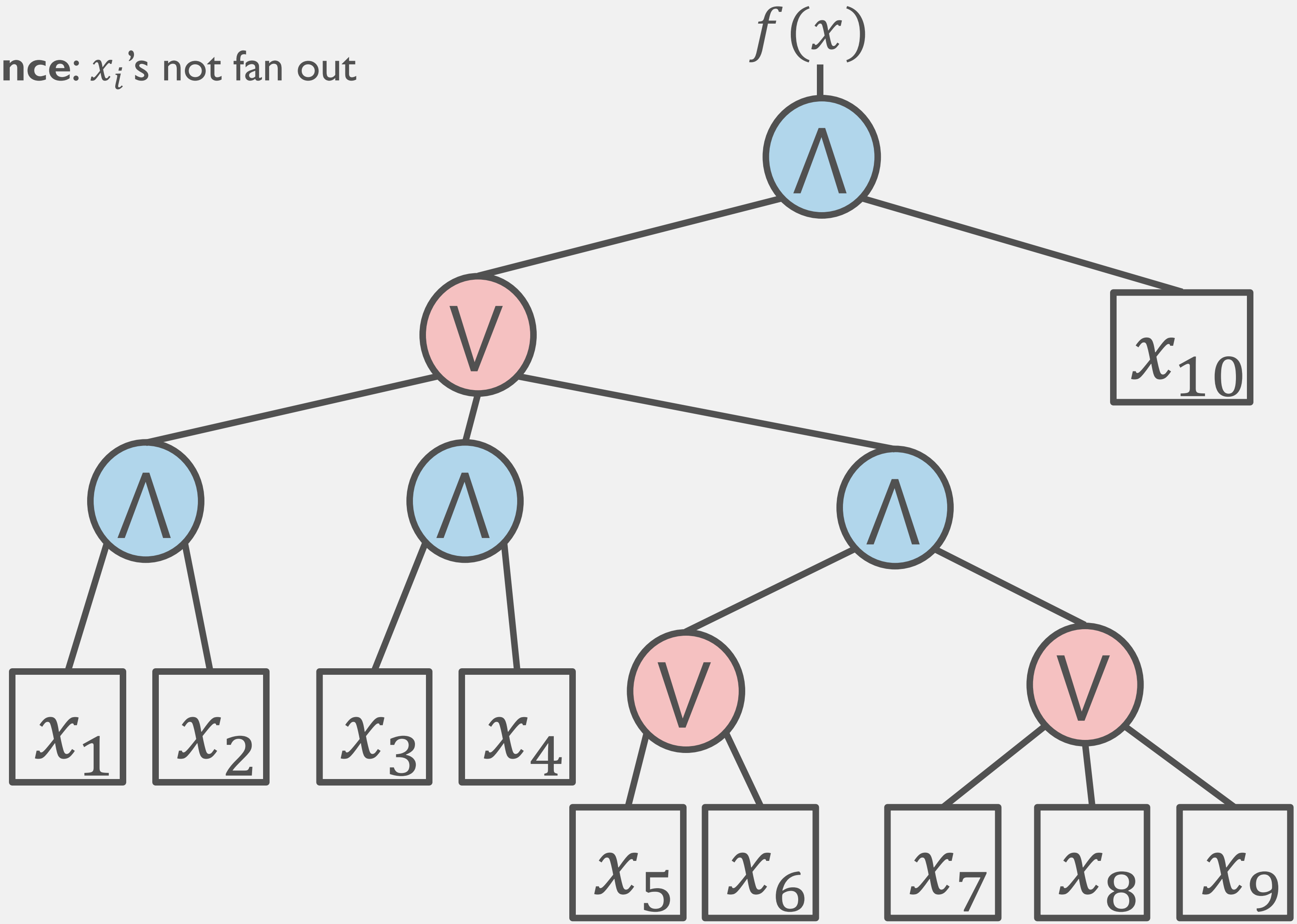


Boolean Formulas



Boolean Formulas

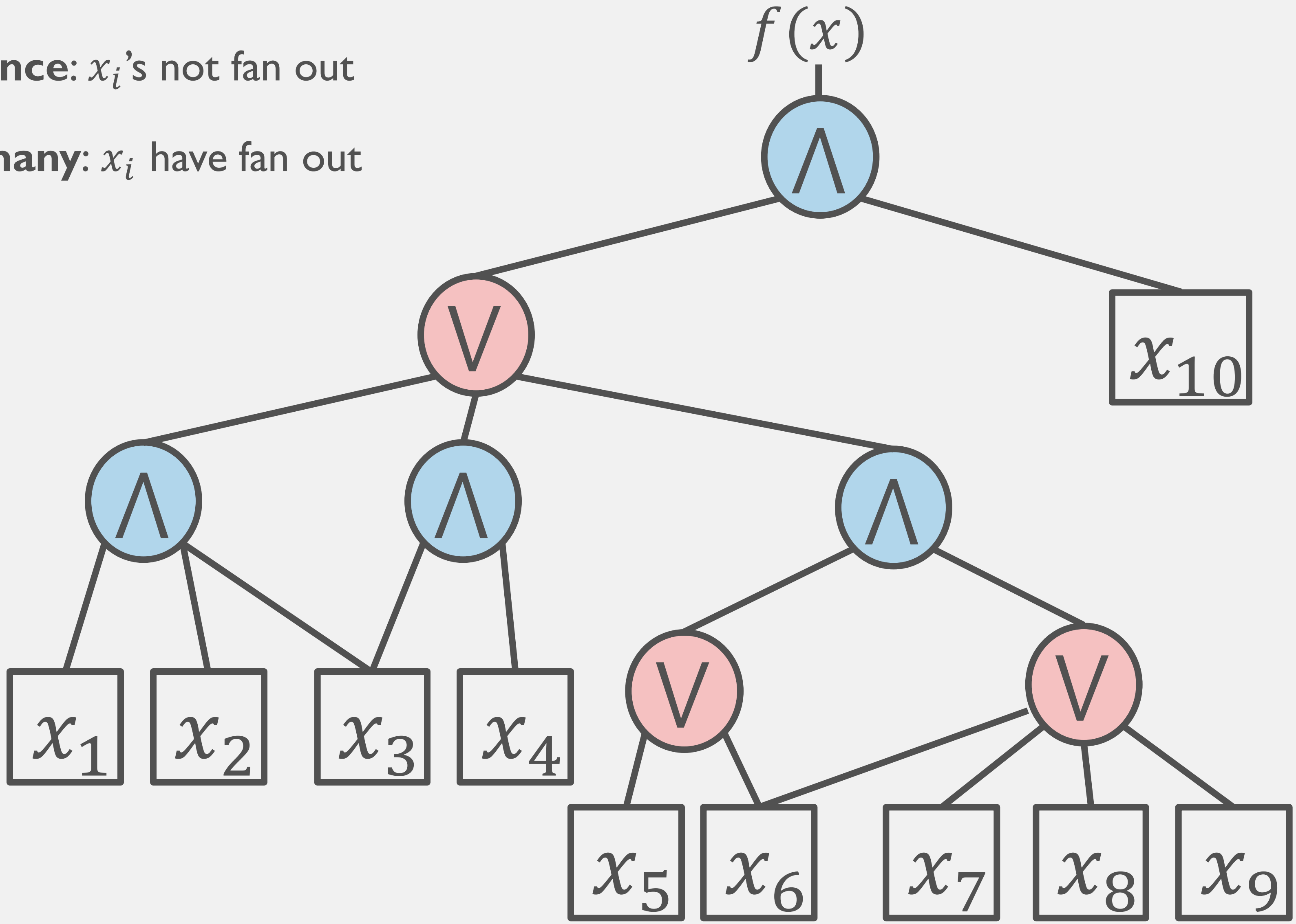
Read-once: x_i's not fan out



Boolean Formulas

Read-once: x_i's not fan out

Read-many: x_i have fan out

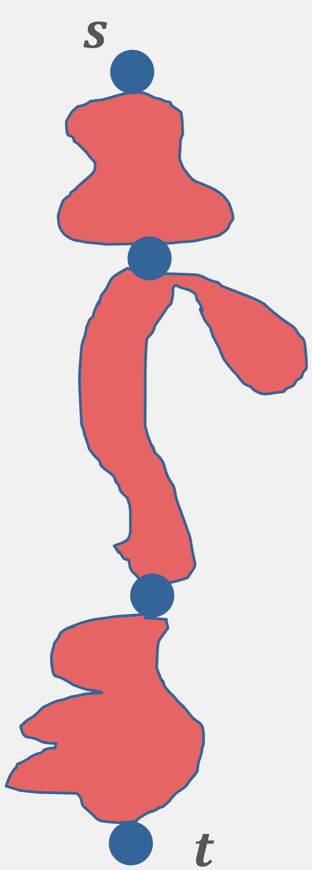


Boolean Formula Applications

- Logic
- Designing electrical circuits
- Game theory (deciding who will win a game)
- Combinatorics and graph problems
- Linear programming
- Testing potential solution to an NP-complete problem

Application to Boolean Formulas

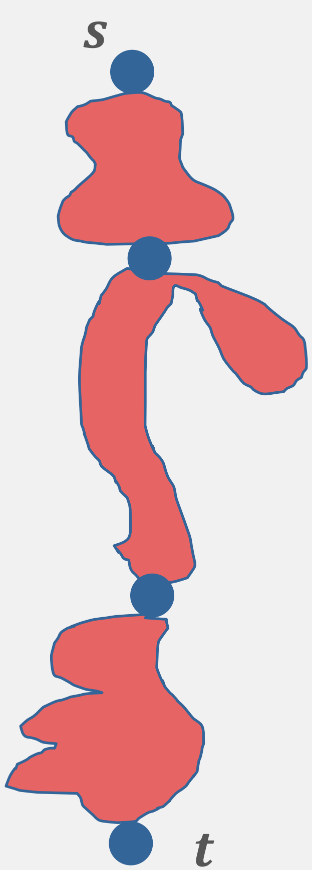
$\wedge$ AND: outputs 1 if all input subformulas have value 1



s and t are connected if all subgraphs are connected

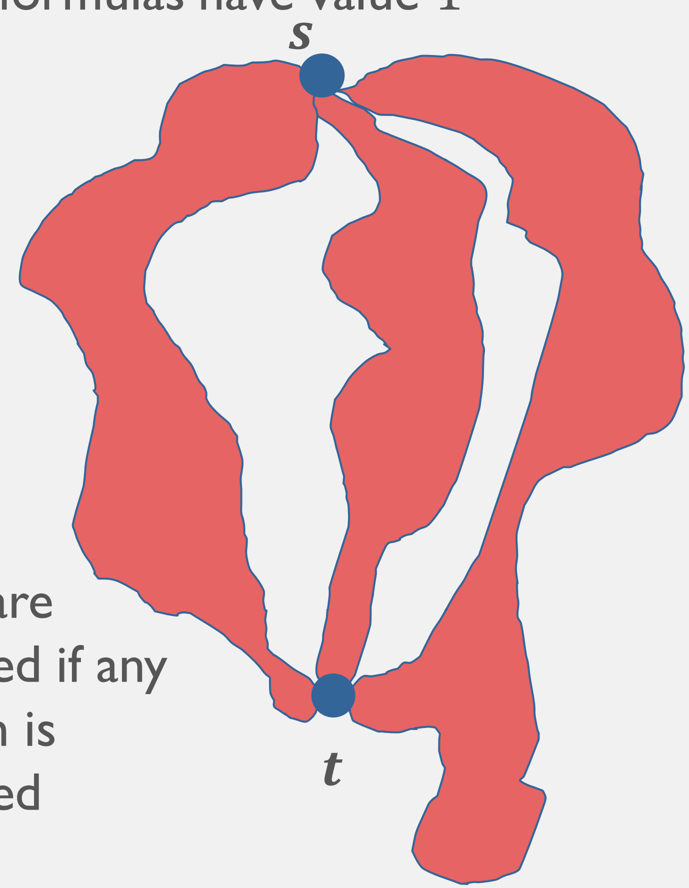
Application to Boolean Formulas

$\wedge$ AND: outputs 1 if all input subformulas have value 1



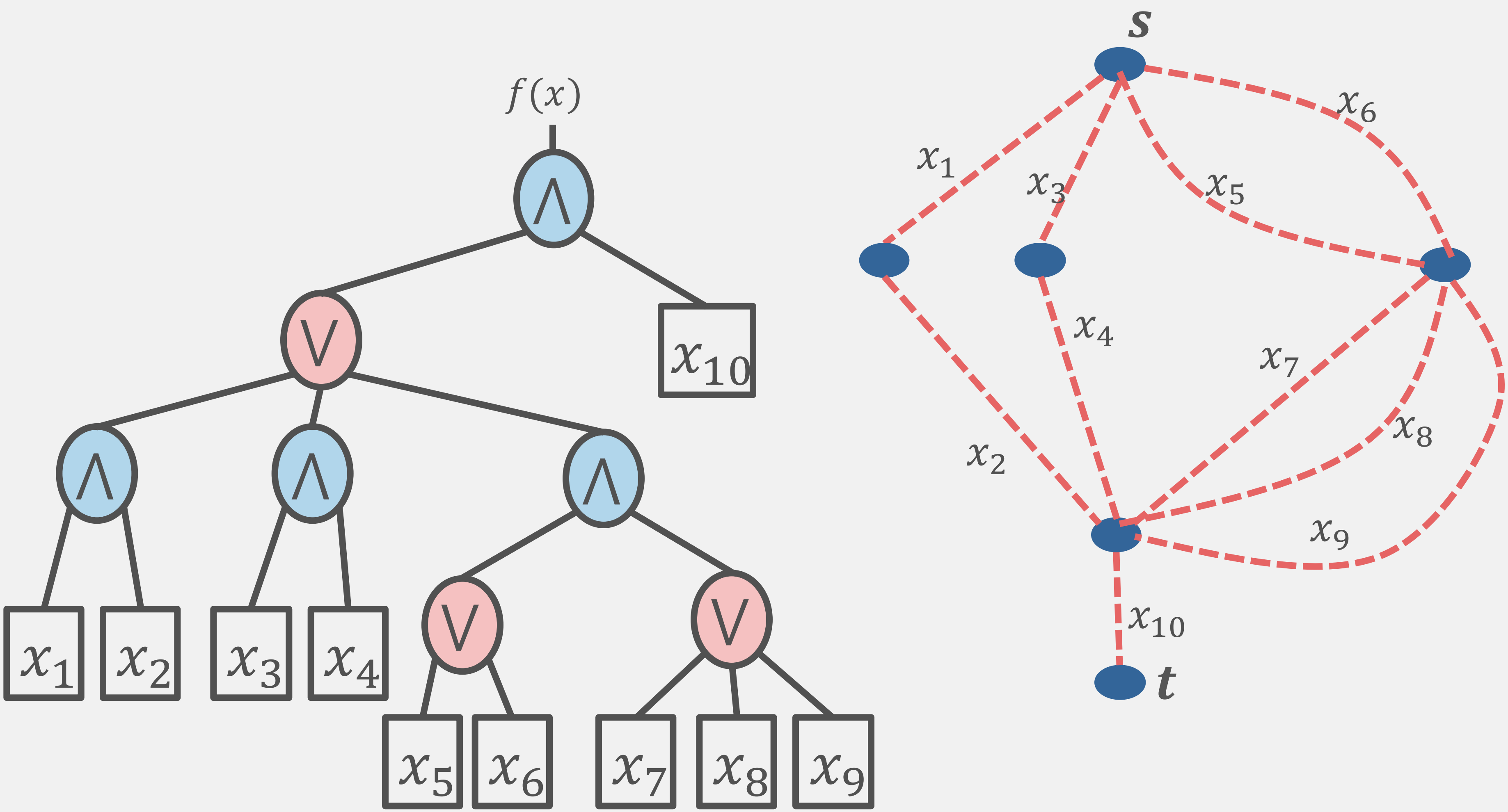
s and t are connected if all subgraphs are connected

$\vee$ OR: outputs 1 if any input subformulas have value 1



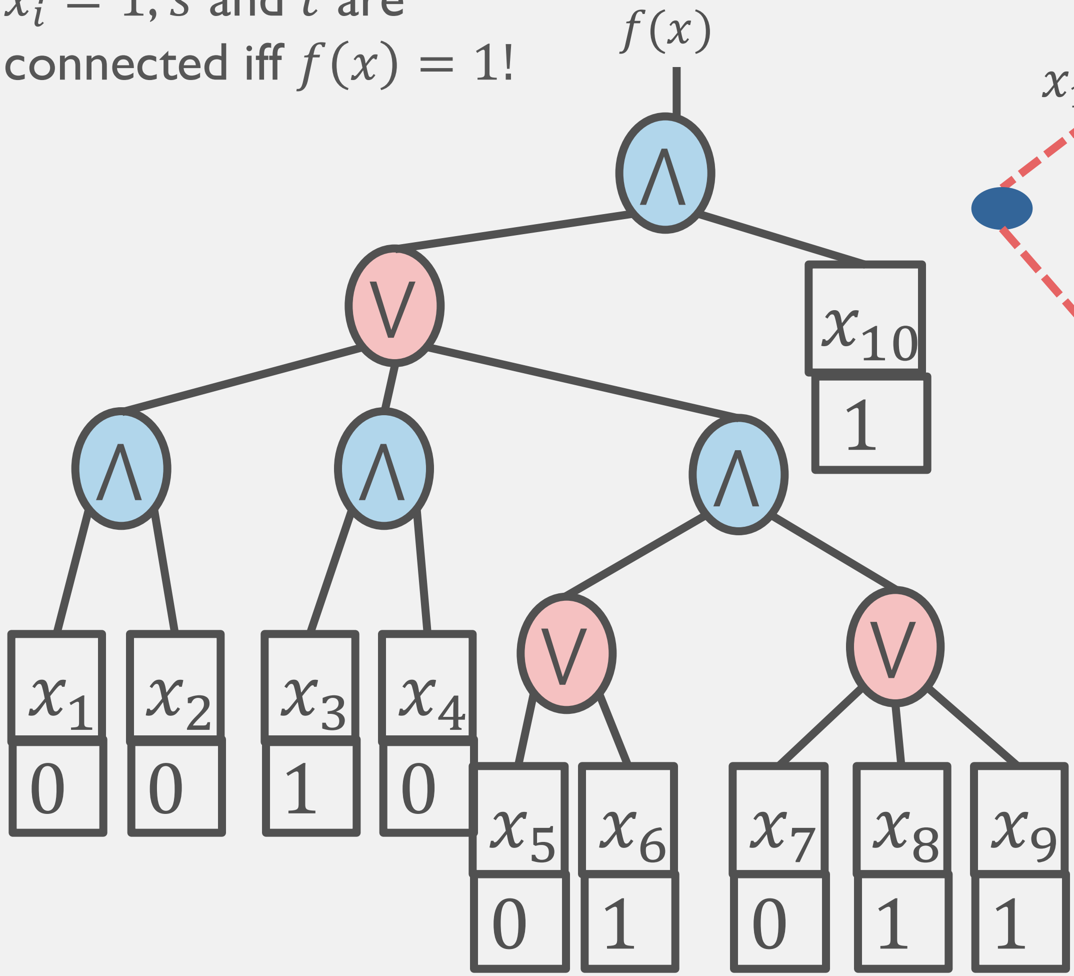
s and t are connected if any subgraph is connected

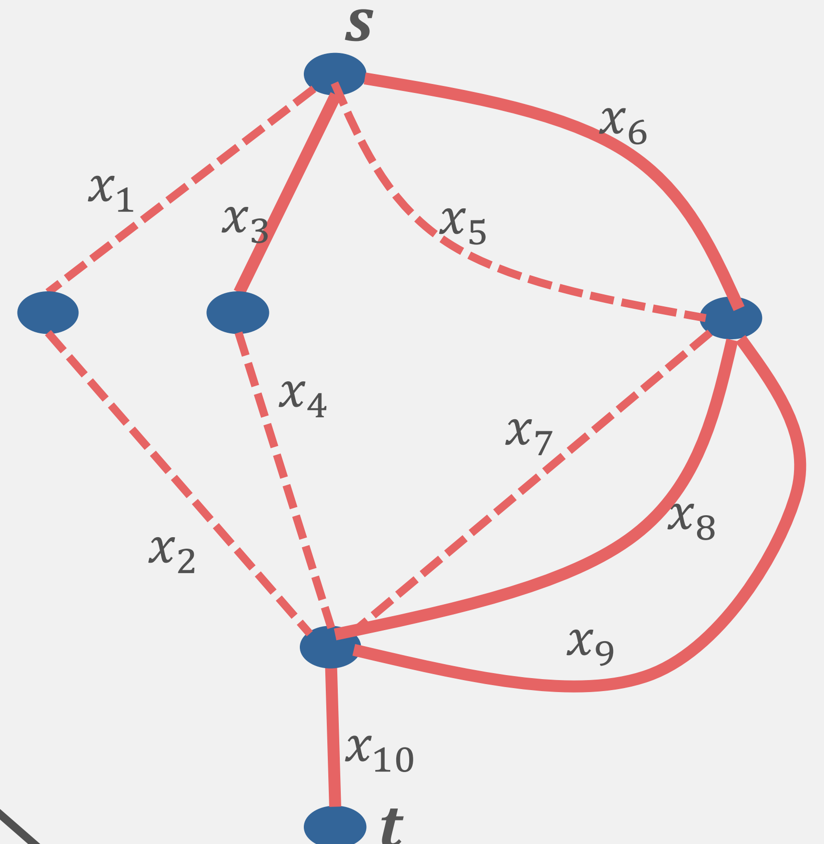
Application to Boolean Formulas



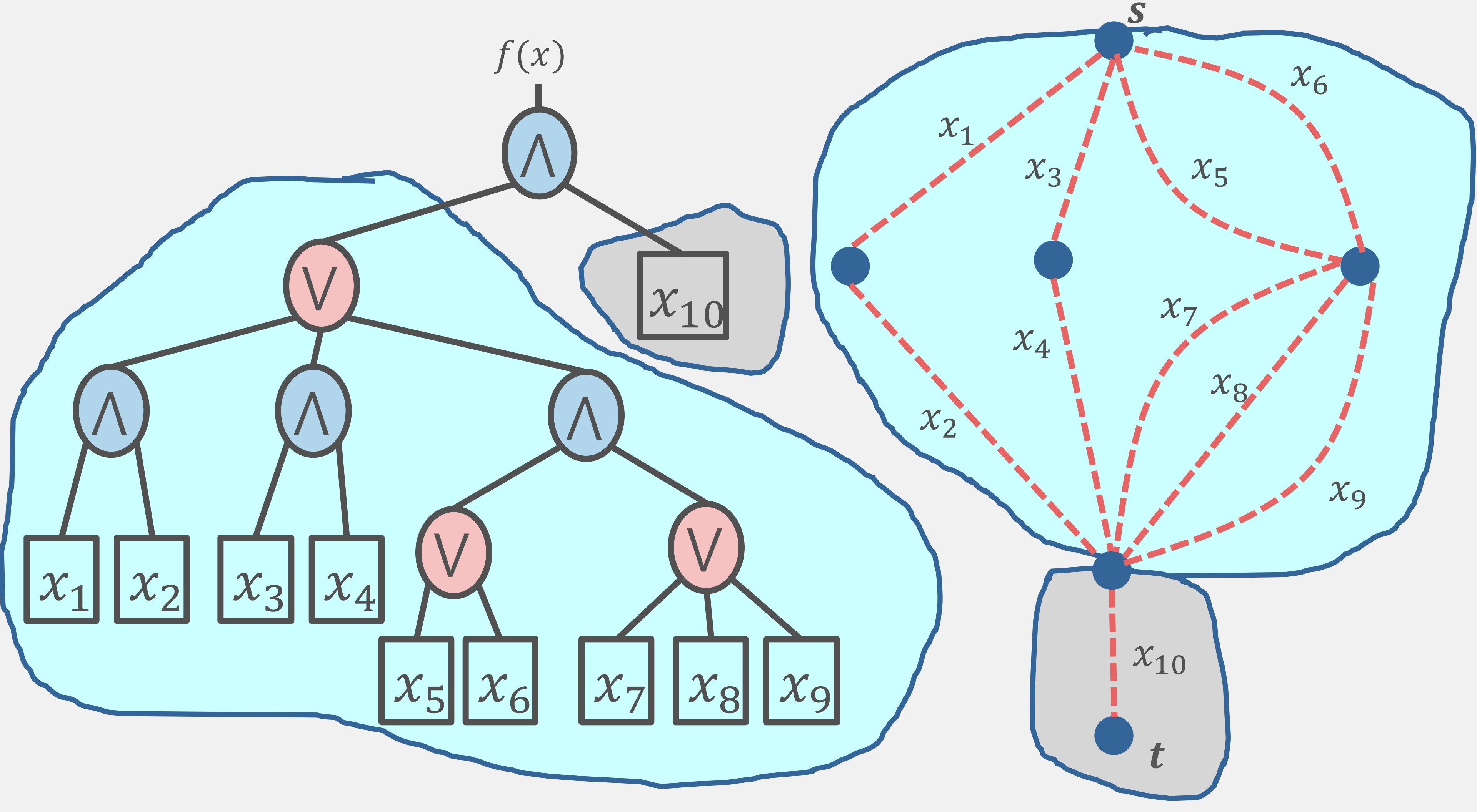
Application to Boolean Formulas

- If we put edges where $x_i = 1$, s and t are connected iff $f(x) = 1$!

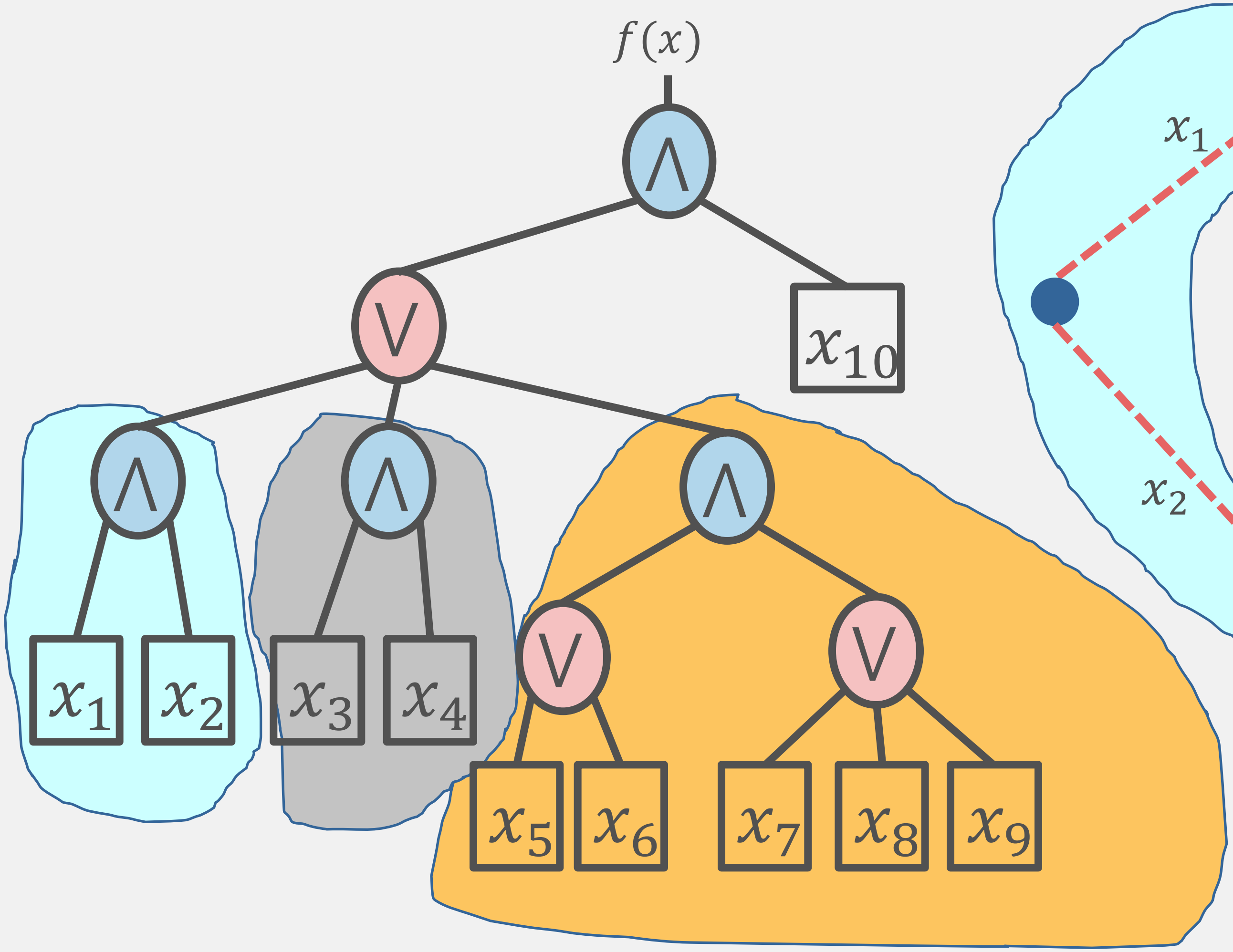


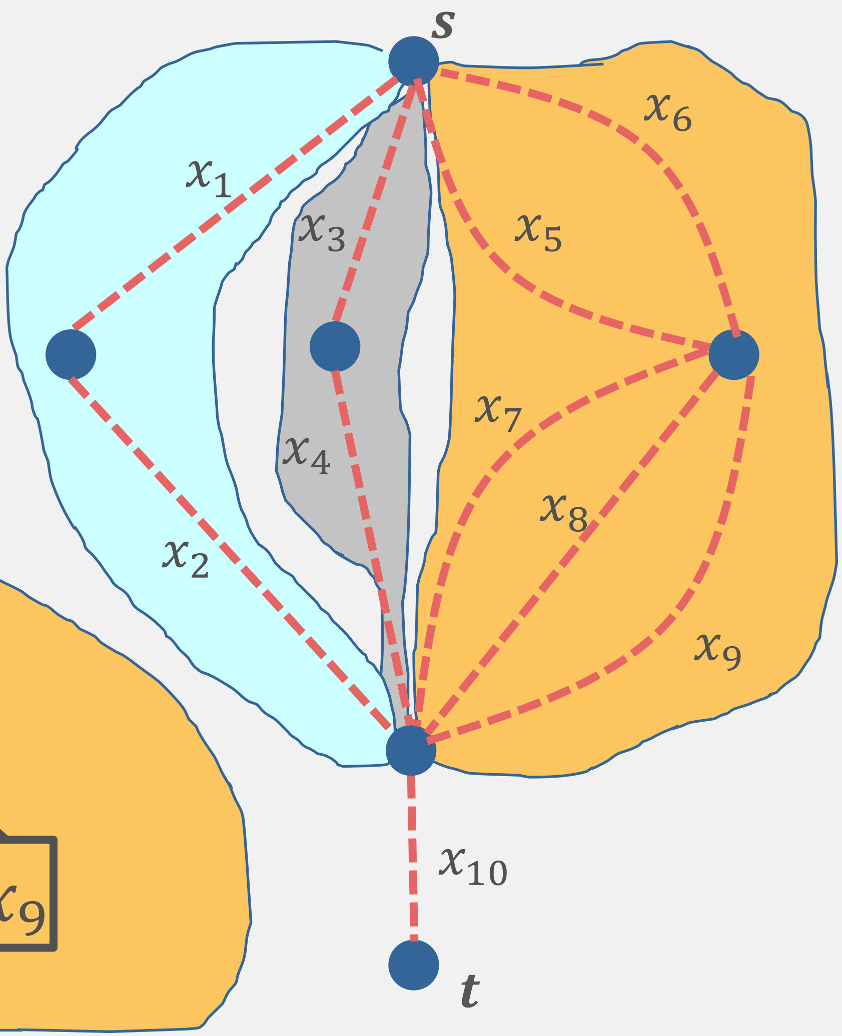


Application to Boolean Formulas



Application to Boolean Formulas



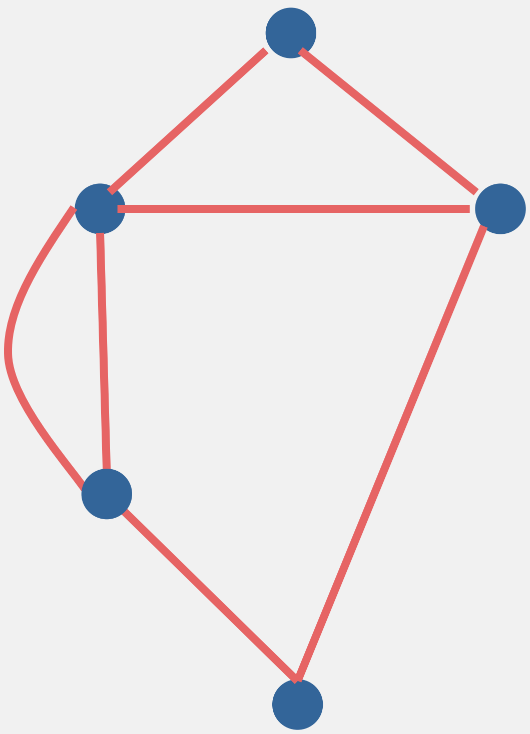


Outline:

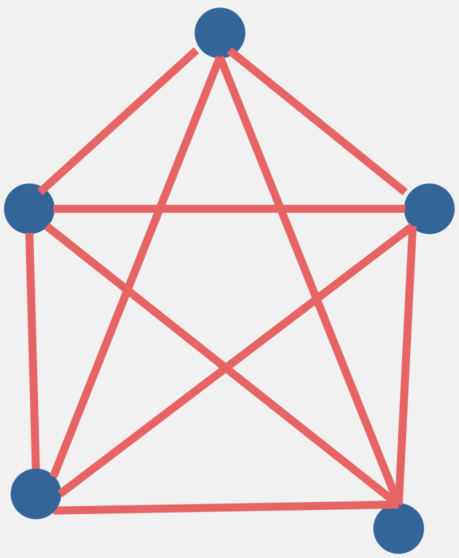
A. Introduction to Quantum Algorithms and st-connectivity
B. st-connectivity makes a good algorithmic primitive
   1. Applies to a wide range of problems
      - Evaluating Boolean formulas reduces to st-connectivity
   2. Easy to understand (without knowing quantum mechanics)

Planar Graph

Planar



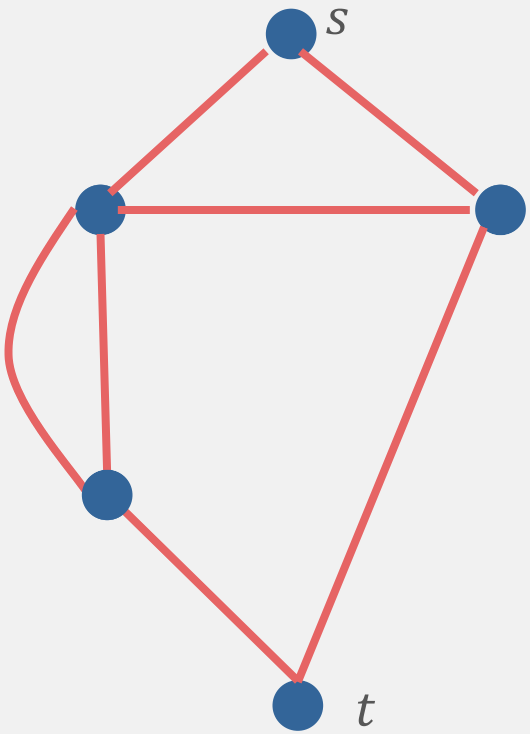
Not Planar



Planar Graph including (s, t) Edge

Can add an edge from s to t and graph is still planar

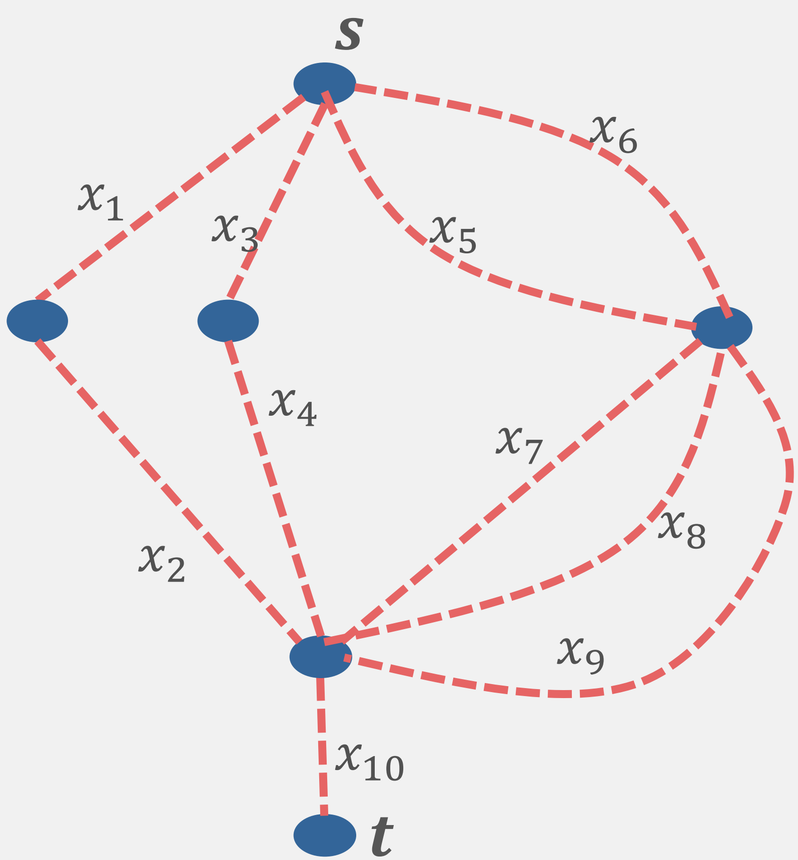
YES



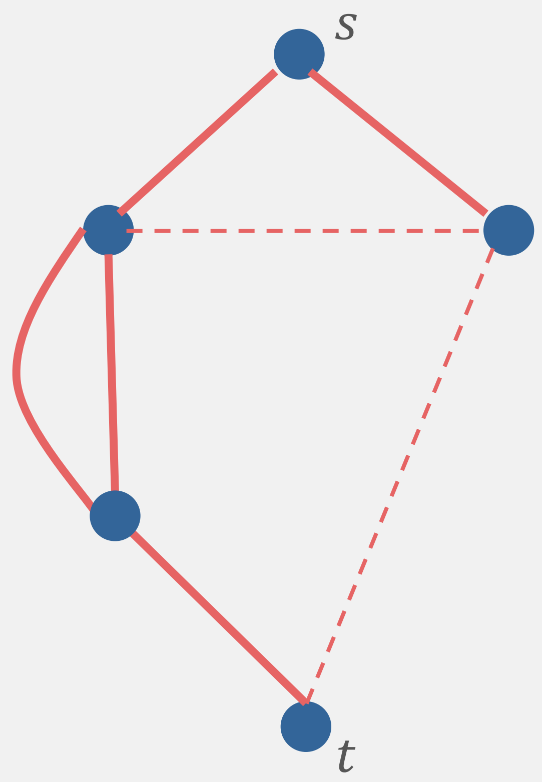
Planar Graph including (s, t) Edge

Can add an edge from s to t and graph is still planar

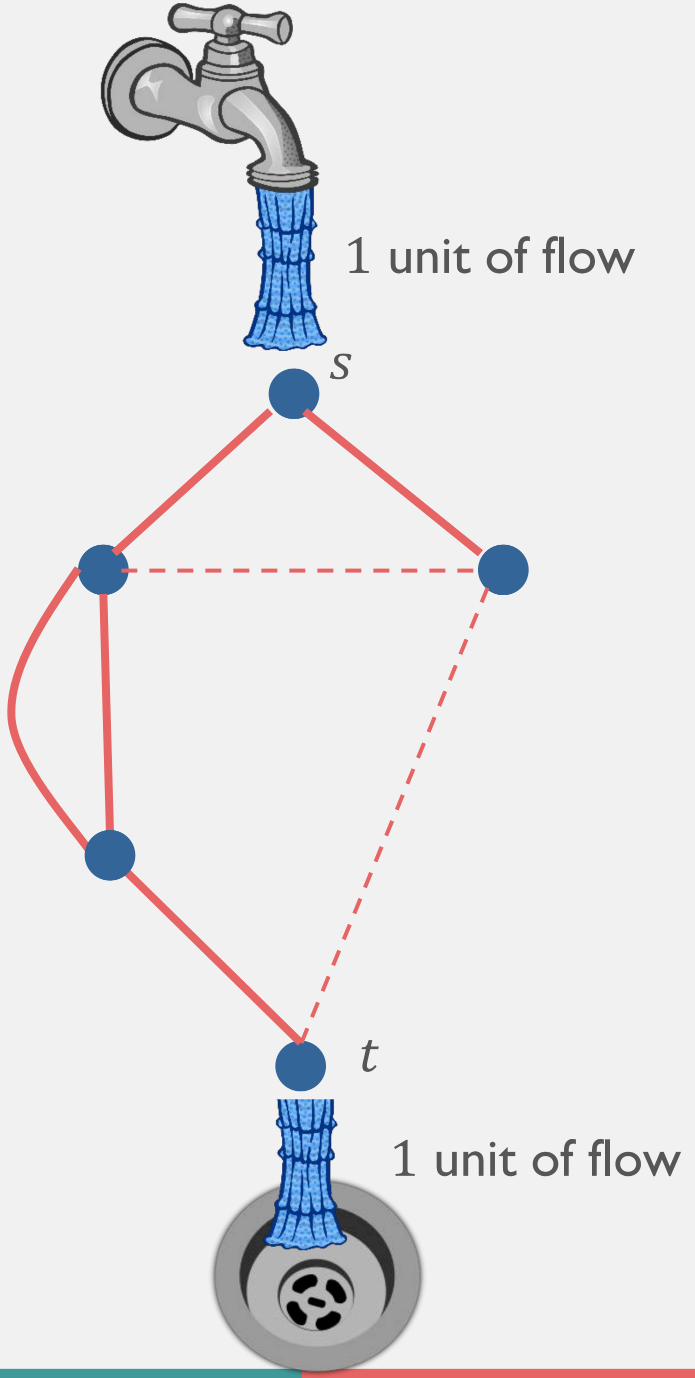
Graph created during reduction from Boolean formula problem has this property by construction.



Effective Resistance



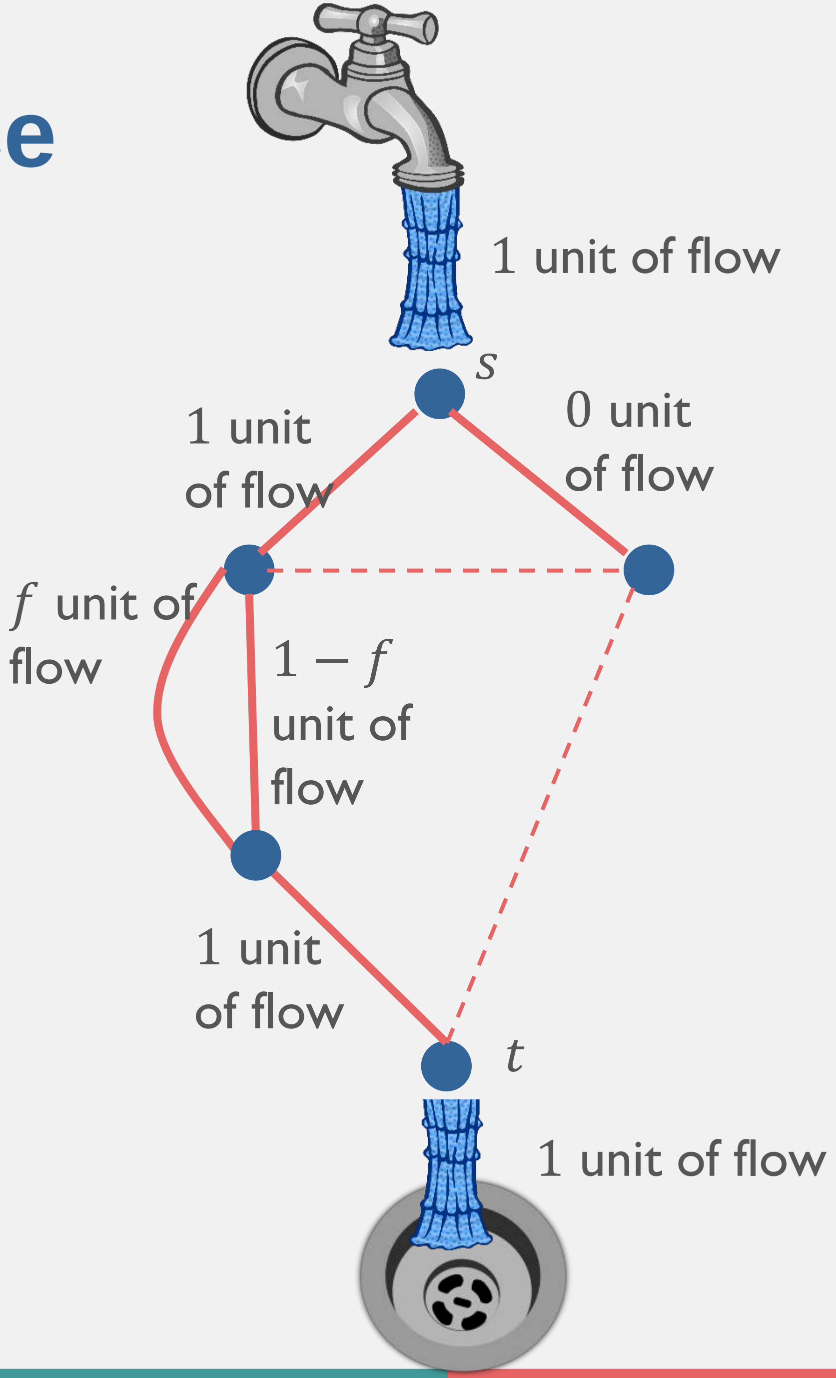
Effective Resistance



Effective Resistance

Valid flow:

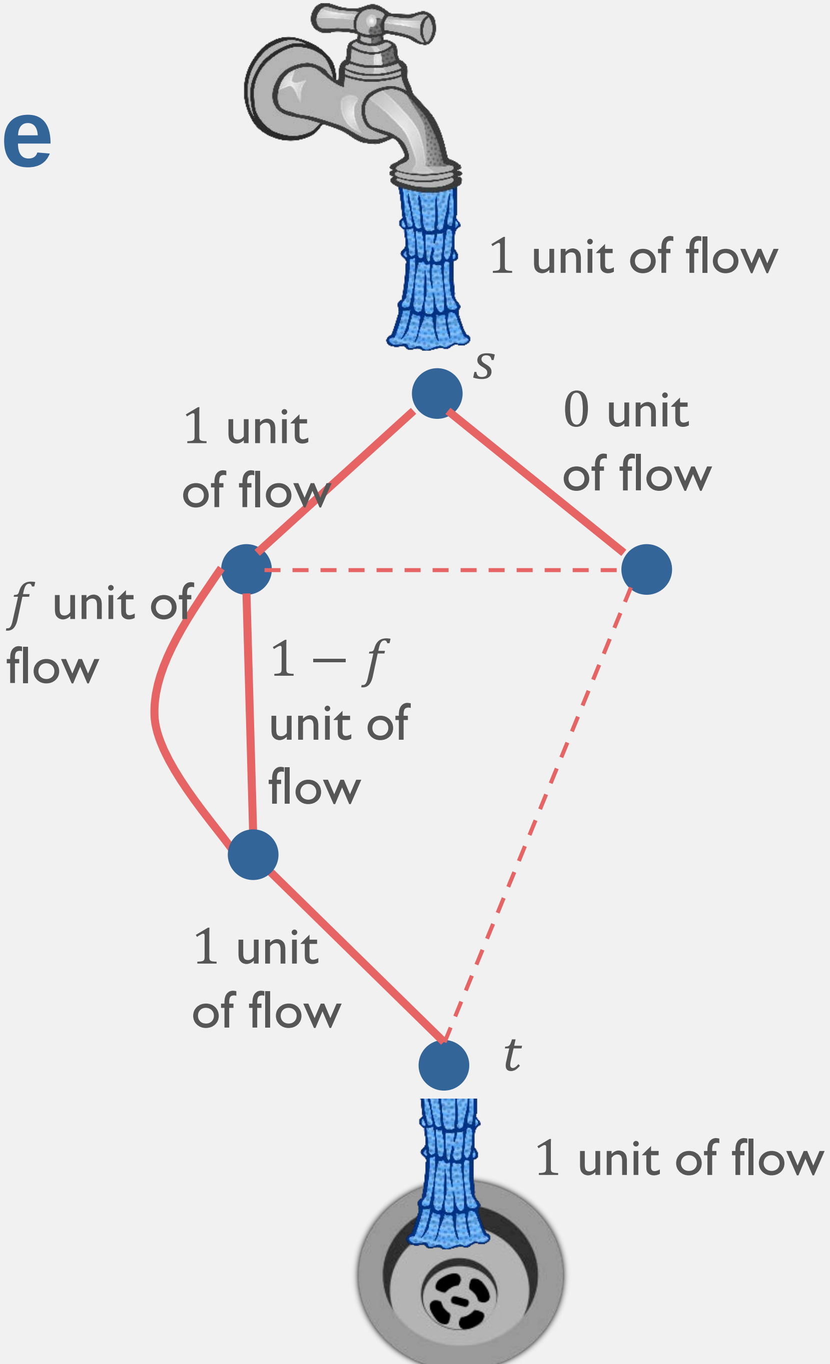
- 1 unit in at s
- 1 unit out at t
- At all other nodes, zero net flow



Effective Resistance

Flow energy:

$$\sum_{edges} (flow\ on\ edge)^2$$

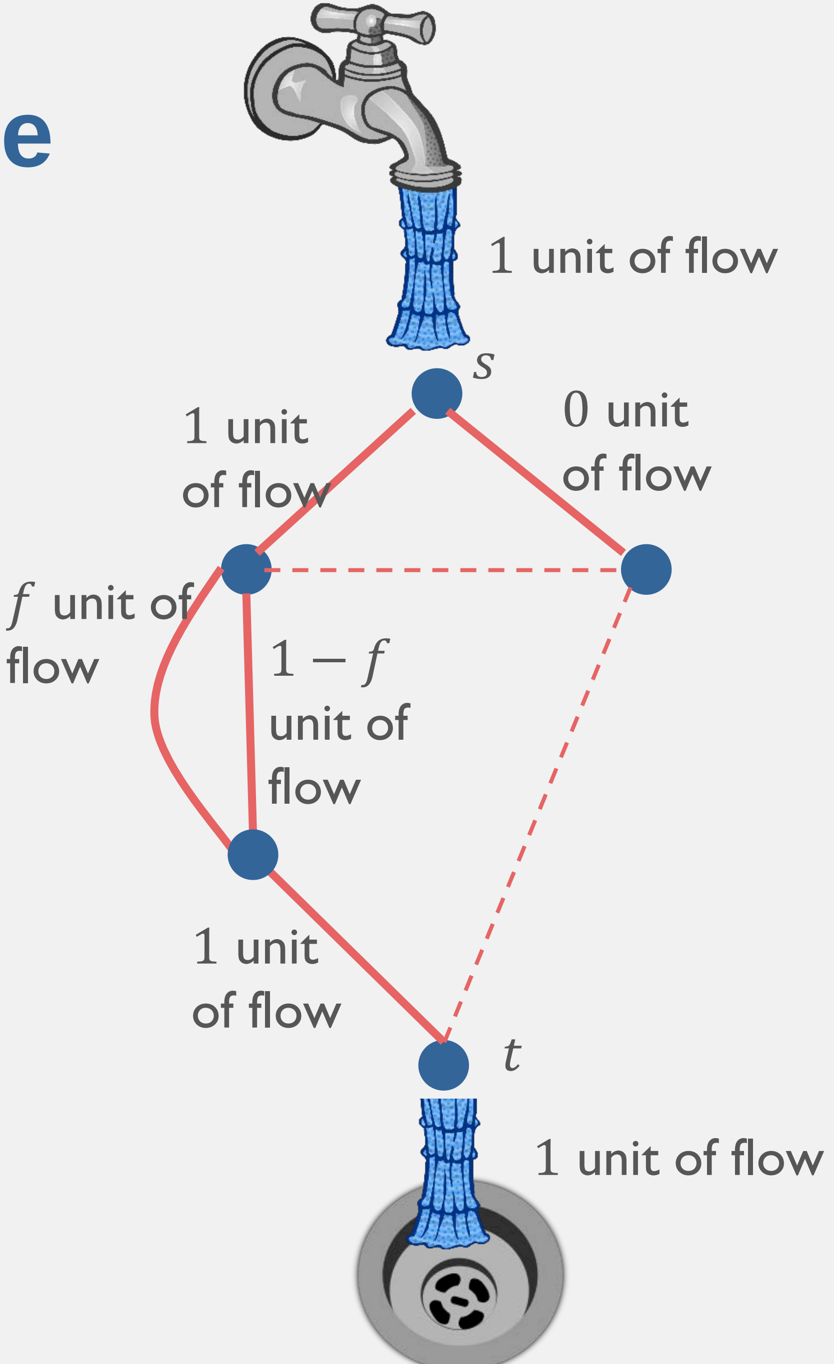


Effective Resistance

Flow energy:

$$\sum_{edges} (flow\ on\ edge)^2$$

Effective Resistance: $R_{s,t}(G)$
Smallest energy of any valid flow from s to t on G.



Effective Resistance

Flow energy:

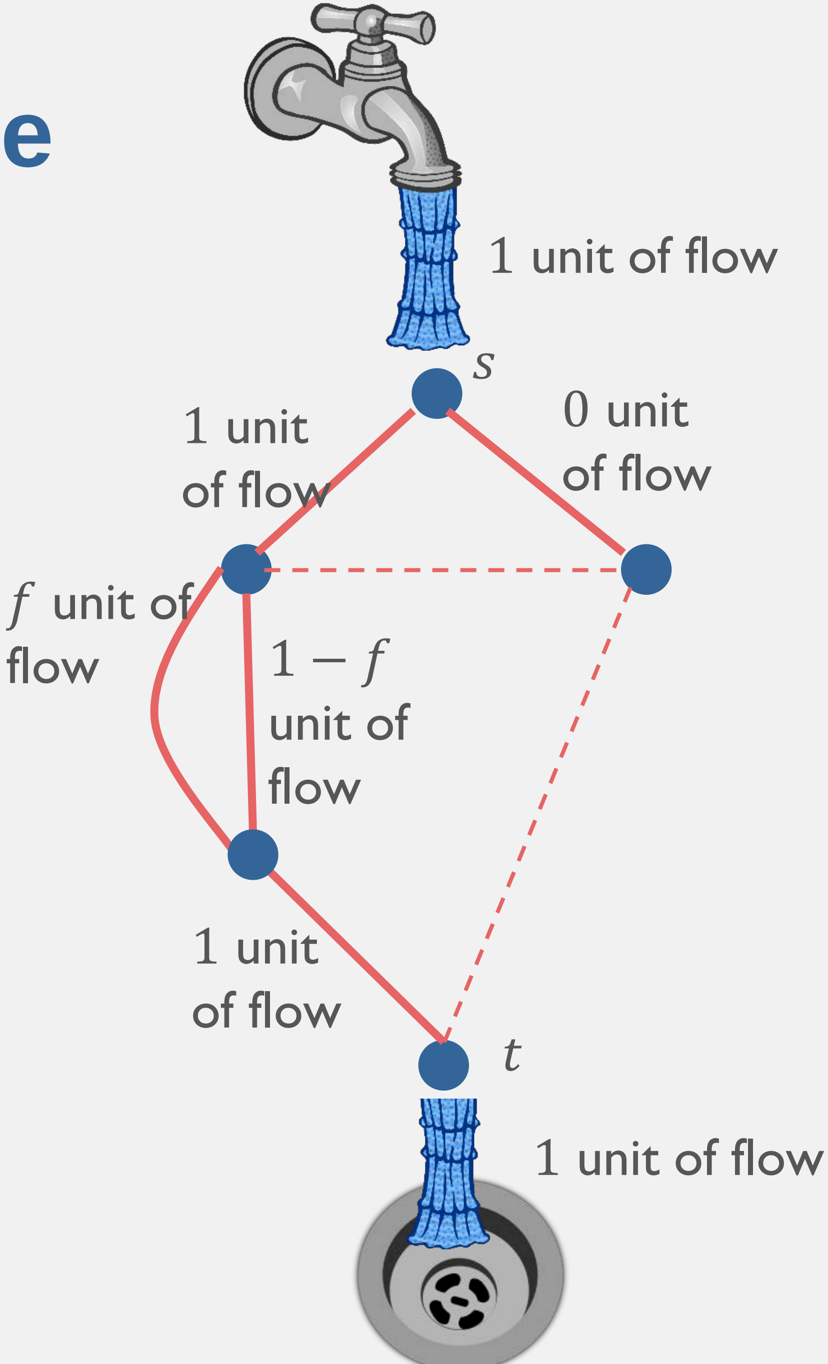
$$\sum_{edges} (flow\ on\ edge)^2$$

Effective Resistance: $R_{s,t}(G)$

Smallest energy of any valid flow from s to t on G.

Properties of $R_{s,t}(G)$

- Small if many short paths from s to t
- Large if few long paths from s to t
- Infinite if s and t not connected



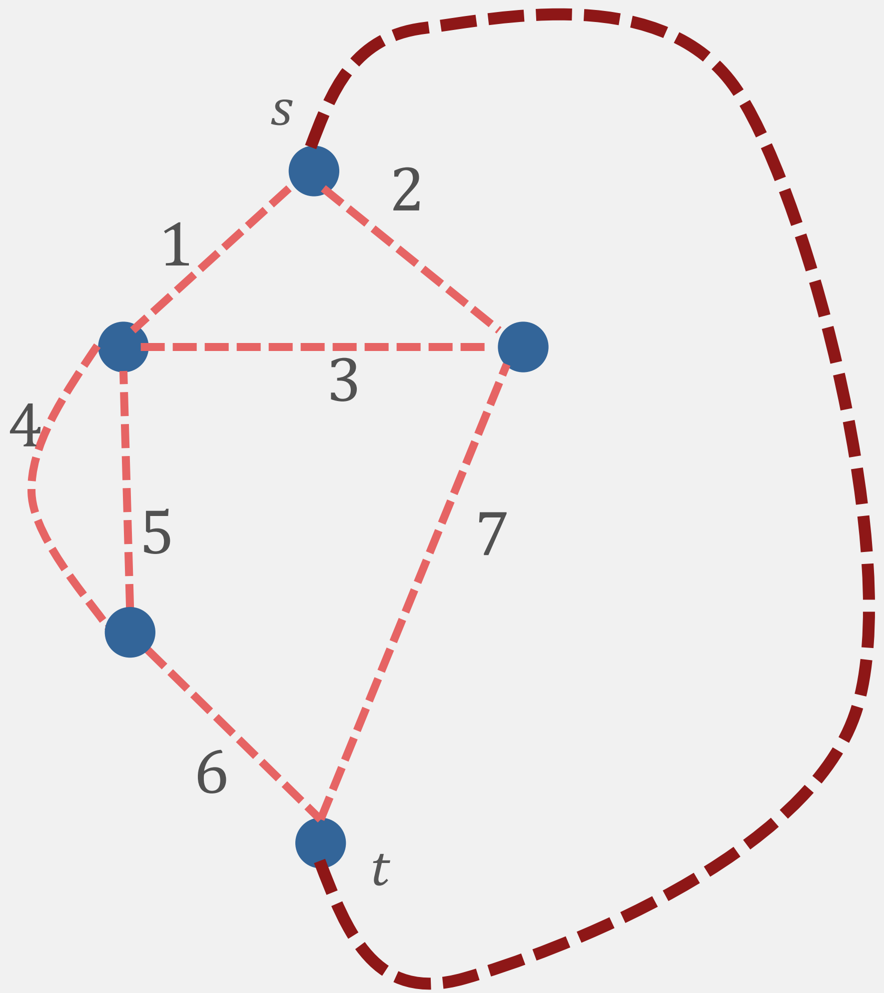
Algorithm Performance:

Planar graph[†] st-connectivity algorithm complexity =

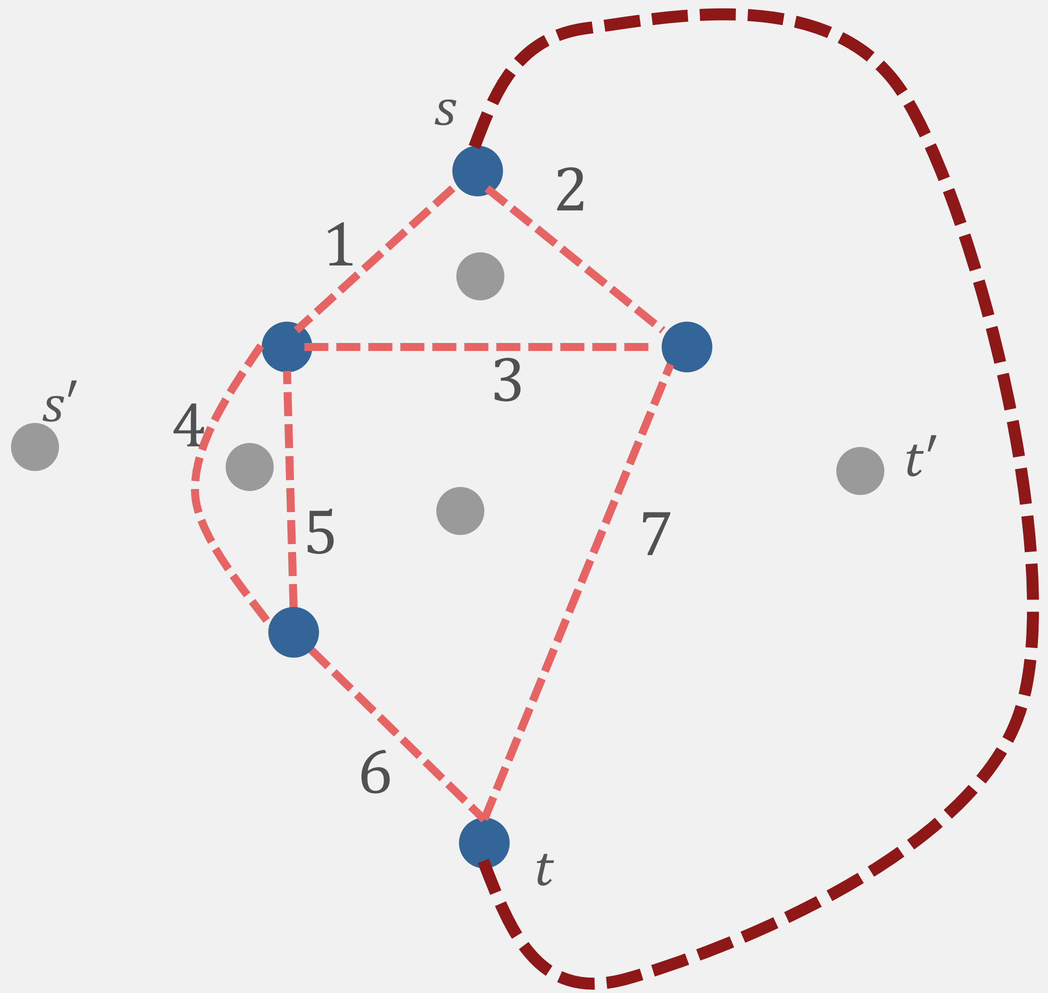
$$O\left(\sqrt{\max_{G\in\mathcal{H}:connected} R_{s,t}(G)}\ \sqrt{\max_{G\in\mathcal{H}:not\ connected} R_{s',t'}(G')}\right)$$

[†] with (s,t) added also planar

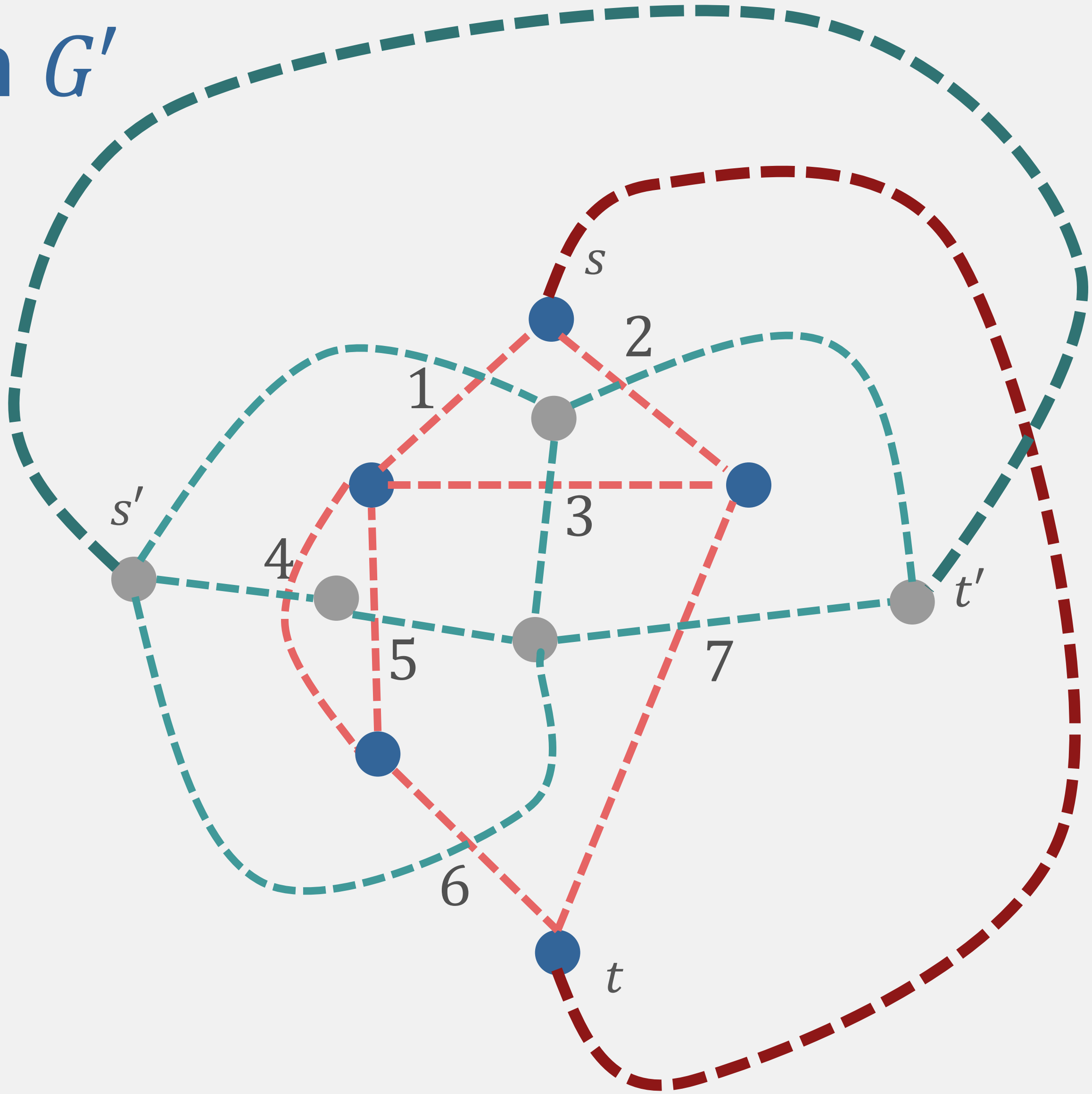
Graph G'



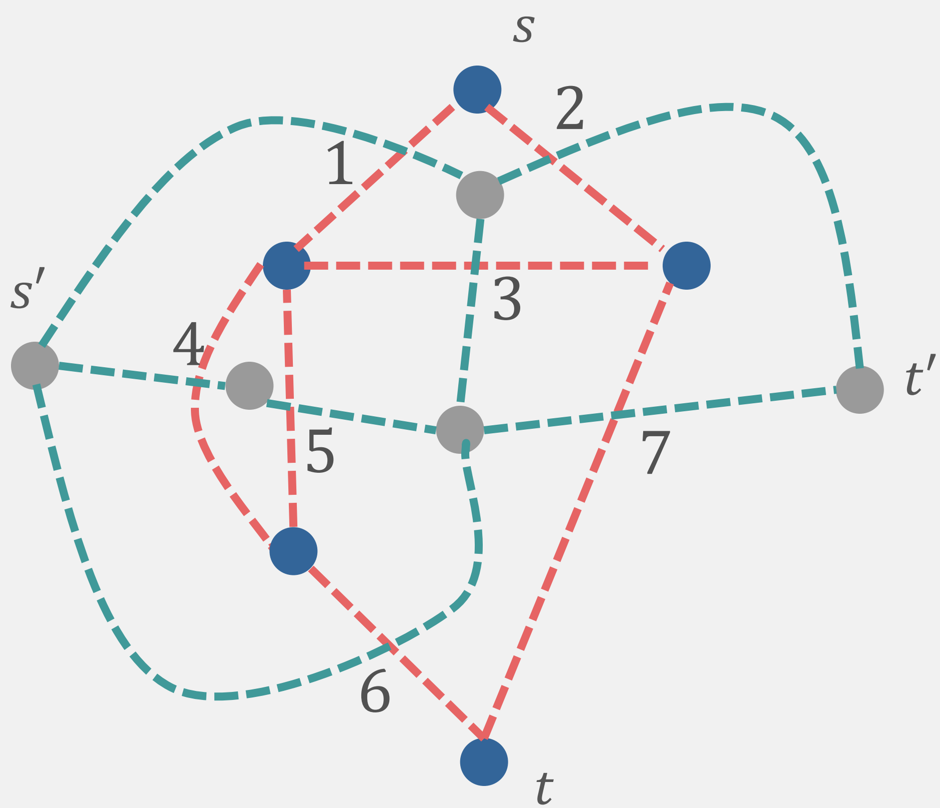
Graph G'



Graph G'

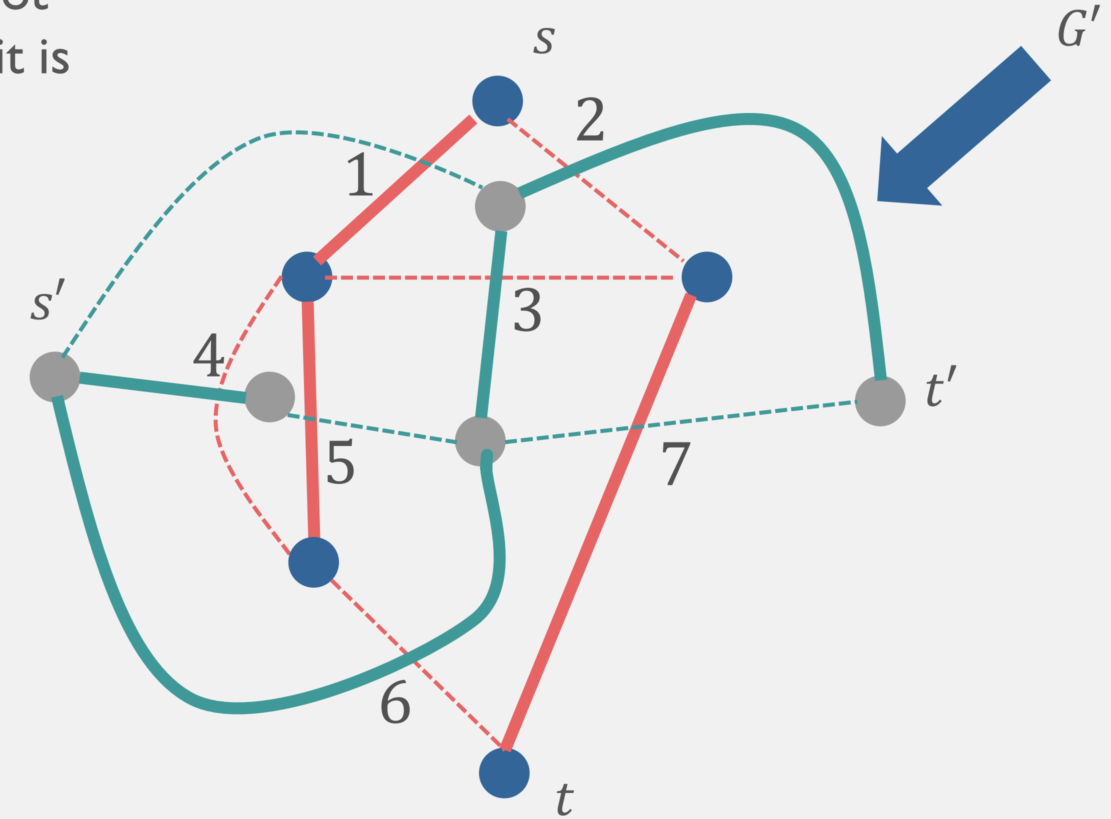


Graph G'



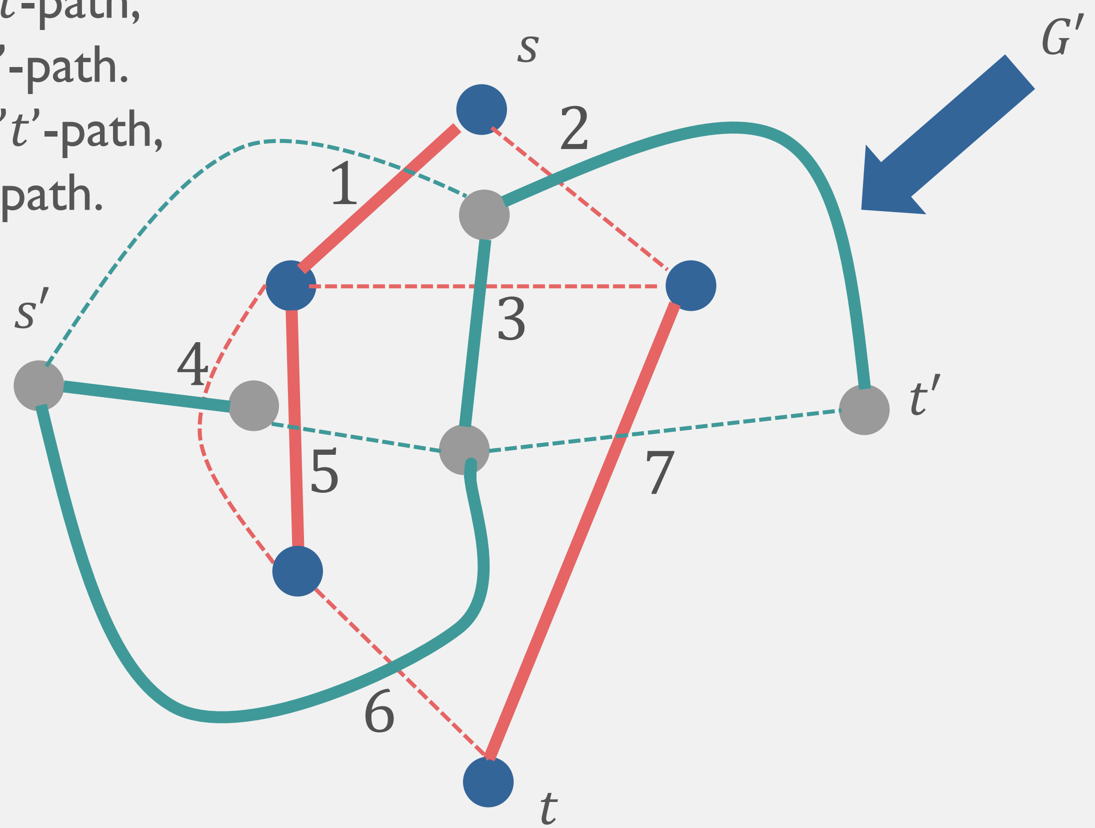
Graph G'

- If an edge is not present in G, it is present in G'



Graph G'

- If there is an st-path, there is no $s't'$-path.
- If there is an $s't'$-path, there is no st-path.



Algorithm Performance:

Planar graph[†] st-connectivity algorithm complexity =

$$O\left(\sqrt{\max_{G\in\mathcal{H}:connected} R_{s,t}(G)}\sqrt{\max_{G\in\mathcal{H}:not\ connected} R_{s',t'}(G')}\right)$$

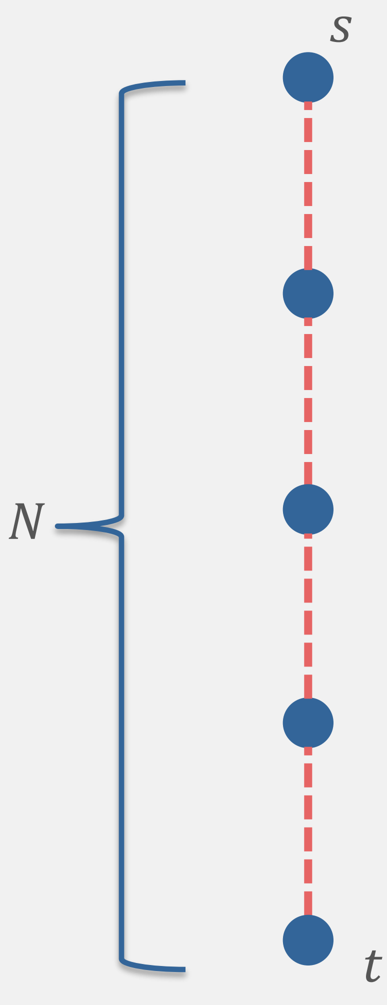
† with (s,t) added also planar

Example

What is quantum complexity of deciding $AND(x_1, x_2, \ldots, x_N)$, promised

- All $x_i = 1$, or
- At least $\sqrt{N}$ input variables are 0.

Example



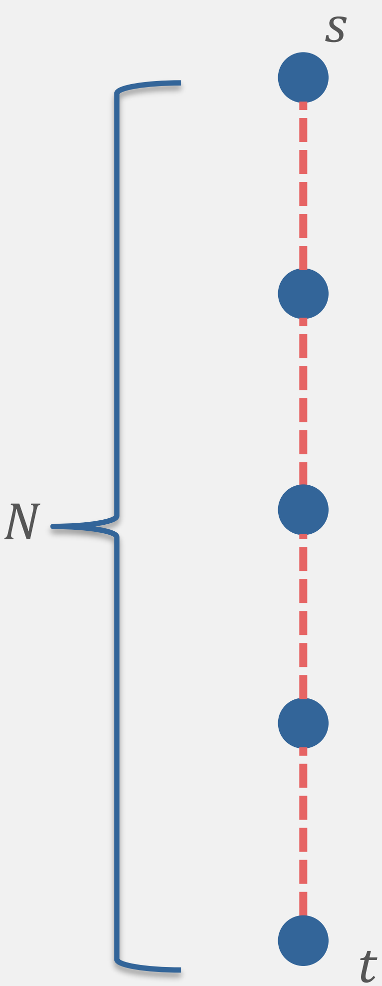
What is quantum complexity of deciding $AND(x_1, x_2, \dots, x_N)$, promised

- All $x_i = 1$, or
- At least $\sqrt{N}$ input variables are 0.

What is quantum complexity of deciding if

- s and t are connected, or
- At least $\sqrt{N}$ edges are missing

Example

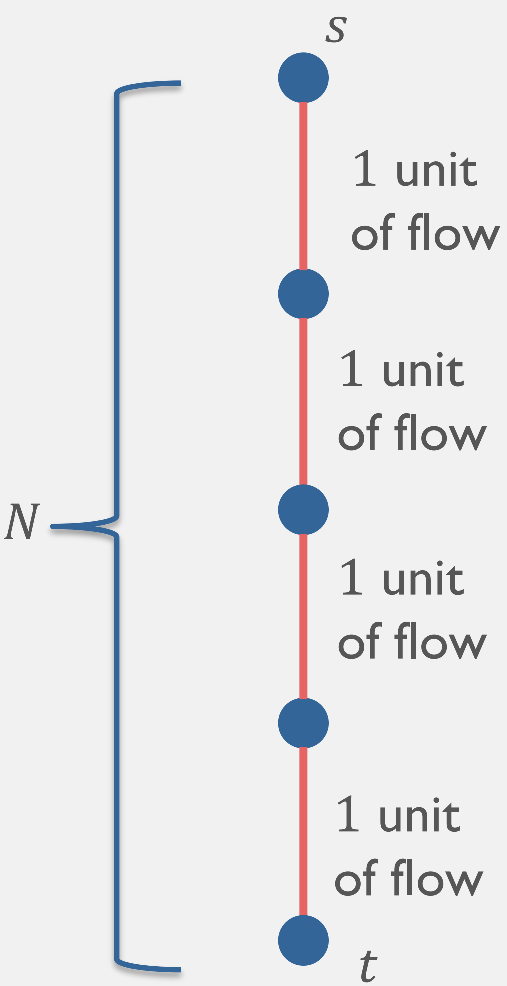


What is quantum complexity of deciding if

- s and t are connected, or
- At least $\sqrt{N}$ edges are missing

$$\sqrt{\max_{G\in\mathcal{H}:connected} R_{s,t}(G)}\ \sqrt{\max_{G\in\mathcal{H}:not\ connected} R_{s',t'}(G')}$$

Example



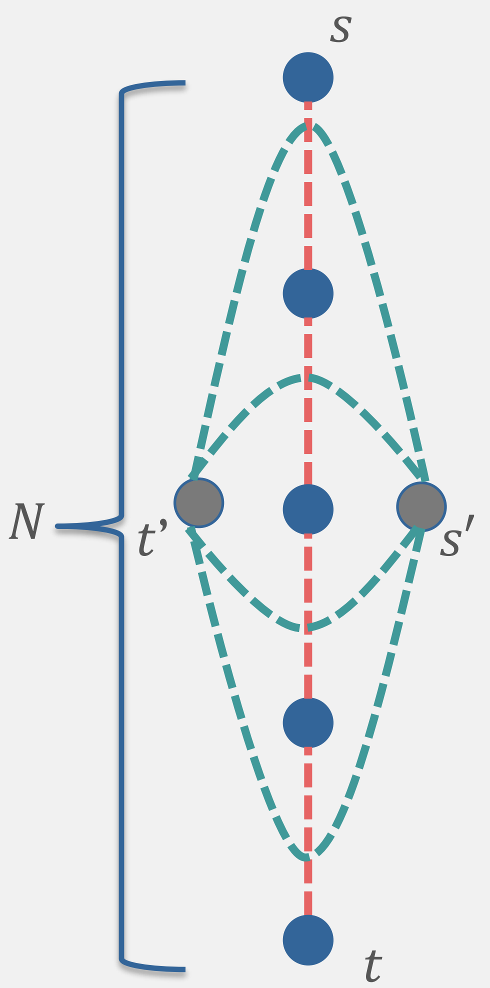
What is quantum complexity of deciding if

- s and t are connected, or
- At least $\sqrt{N}$ edges are missing

$$\sqrt{\max_{G\in\mathcal{H}:connected} R_{s,t}(G)}\ \sqrt{\max_{G\in\mathcal{H}:not\ connected} R_{s',t'}(G')}$$

$$\max_{G\in\mathcal{H}:connected} R_{s,t}(G) = N$$

Example

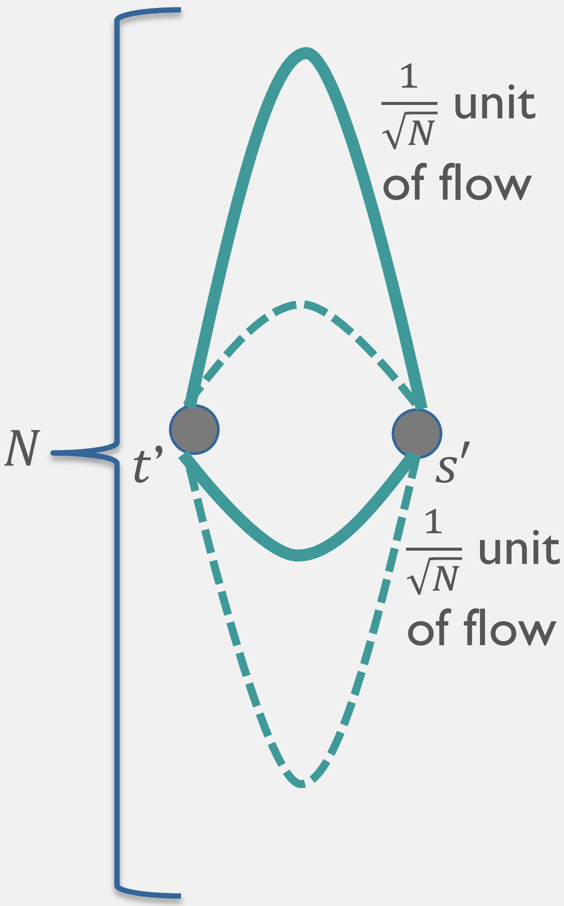


What is quantum complexity of deciding if

- s and t are connected, or
- At least $\sqrt{N}$ edges are missing

$$\sqrt{\max_{G\in\mathcal{H}:connected} R_{s,t}(G)}\ \sqrt{\max_{G\in\mathcal{H}:not\ connected} R_{s',t'}(G')}$$

Example



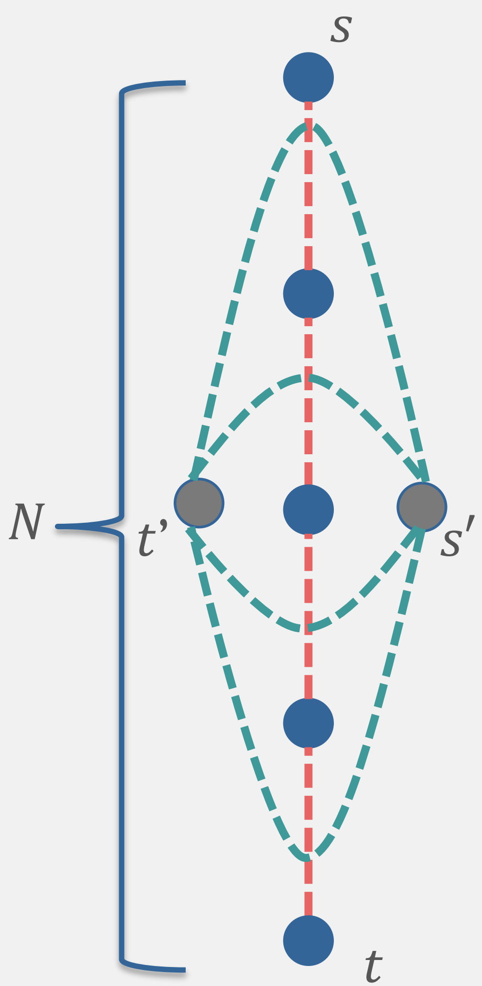
What is quantum complexity of deciding if

- s and t are connected, or
- At least $\sqrt{N}$ edges are missing

$$\sqrt{\max_{G \in \mathcal{H}:connected} R_{s,t}(G)} \sqrt{\max_{G \in \mathcal{H}:not\ connected} R_{s',t'}(G')}$$

$$\max_{G \in \mathcal{H}:not\ connected} R_{s,t}(G') = 1/\sqrt{N}$$

Example



What is quantum complexity of deciding if

- s and t are connected, or
- At least $\sqrt{N}$ edges are missing

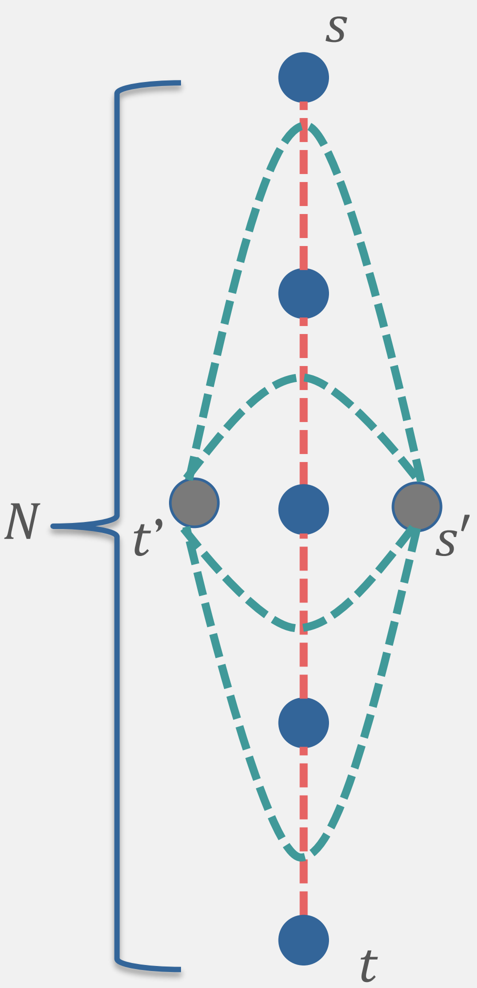
$$\sqrt{\max_{G\in\mathcal{H}:connected} R_{s,t}(G)} \qquad \sqrt{\max_{G\in\mathcal{H}:not\ connected} R_{s',t'}(G')}$$

$$\Big\downarrow \qquad\qquad \Big\downarrow$$

$$N \qquad\qquad 1/\sqrt{N}$$

Quantum complexity is $O\left(N^{1/4}\right)$

Example



What is quantum complexity of deciding if

- s and t are connected, or
- At least $\sqrt{N}$ edges are missing

$$\sqrt{\max_{G \in \mathcal{H}:connected} R_{s,t}(G)} \quad \sqrt{\max_{G \in \mathcal{H}:not\ connected} R_{s',t'}(G')}$$

$$\downarrow \qquad\qquad \downarrow$$

$$N \qquad\qquad 1/\sqrt{N}$$

Quantum complexity is $O\left(N^{1/4}\right)$

Randomized classical complexity is $\Omega\left(N^{1/2}\right)$

Algorithm Performance:

Planar graph[†] st-connectivity algorithm complexity =

$$O\left(\sqrt{\max_{G\in\mathcal{H}:connected} R_{s,t}(G,w)}\sqrt{\max_{G\in\mathcal{H}:not\ connected} R_{s',t'}(G',w)}\right)$$

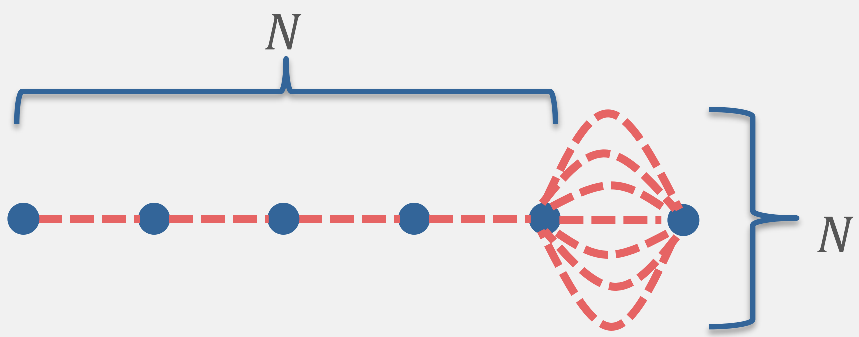
[†] with (s,t) added also planar

Performance

- Improvement over previous quantum st −connectivity algorithm
  - Find a family of graphs with N edges such that our algorithm uses $O(1)$ queries, previous best algorithm uses $O\left(N^{1/4}\right)$ queries

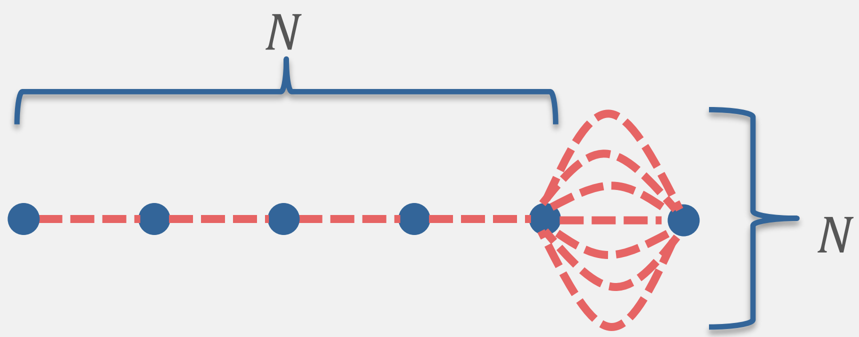
Performance

- Improvement over previous quantum st −connectivity algorithm
  - Find a family of graphs with N edges such that our algorithm uses $O(1)$ queries, previous best algorithm uses $O\left(N^{1/4}\right)$ queries
  - Balloon graph: our algorithm uses $O\left(N^{1/2}\right)$ queries, previous best algorithm uses $O(N)$ queries



Performance

- Improvement over previous quantum st −connectivity algorithm
  - Find a family of graphs with N edges such that our algorithm uses $O(1)$ queries, previous best algorithm uses $O\left(N^{1/4}\right)$ queries
  - Balloon graph: our algorithm uses $O\left(N^{1/2}\right)$ queries, previous best algorithm uses $O(N)$ queries



  - Series-parallel graphs, our algorithm uses $O\left(N^{1/2}\right)$ queries, previous best algorithm uses $O(N)$ queries

Performance

- Comparison to previous Boolean formula algorithm
  - Match celebrated result that $O(\sqrt{N})$ queries required for total read-once Boolean formulas, but proof is simple!
  - Extend super-polynomial quantum to classical speed-up for families of NAND-trees [Z**K**H'12, **K**'13]

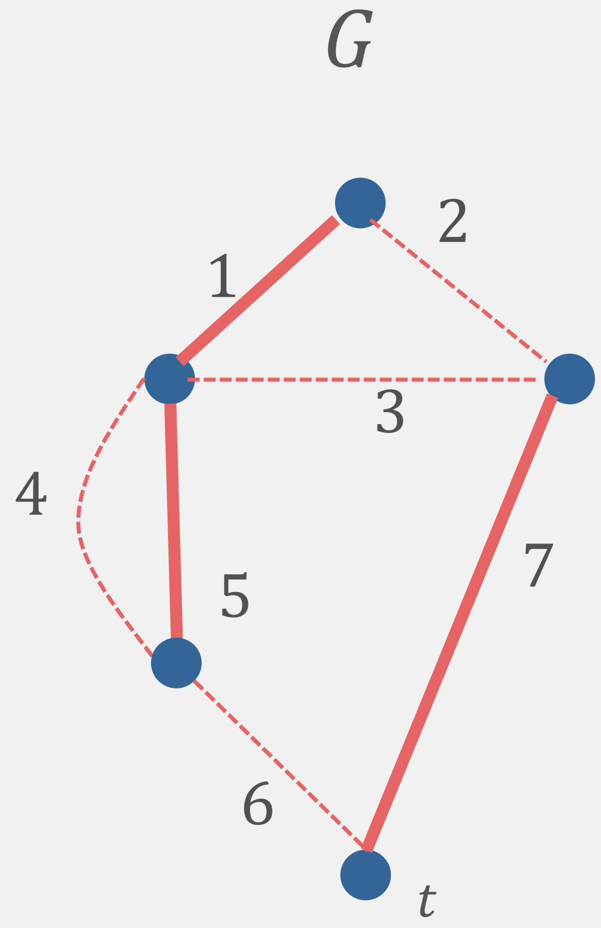
Update

Non-planar st-connectivity algorithm complexity =

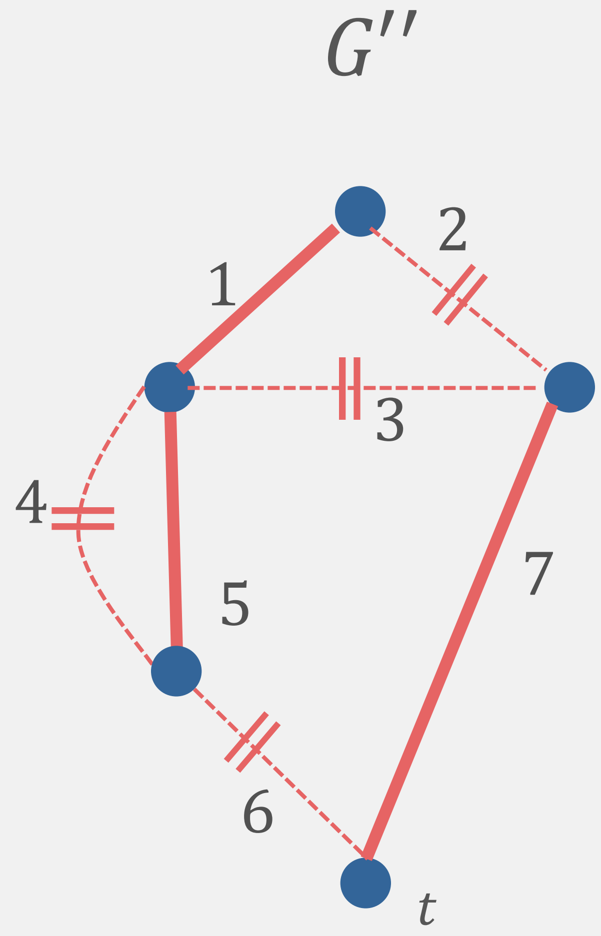
$$O\left(\sqrt{\max_{G\in\mathcal{H}:connected} R_{s,t}(G,w)}\ \sqrt{\max_{G\in\mathcal{H}:not\ connected} C_{s,t}(G'',w)}\right)$$

Effective capacitance

Update



Update



Open Questions

- When is our algorithm optimal for Boolean formulas? (Especially partial/read-many formulas)
- Are there other problems that reduce to st-connectivity?
- What is the classical time/query complexity of st-connectivity in the black box model?
- Does our reduction from formulas to connectivity give good classical algorithms too?
- Can we use this graph dual idea to improve other quantum algorithms?

arXiv:1704.00765, with Stacey Jeffery

Other interests

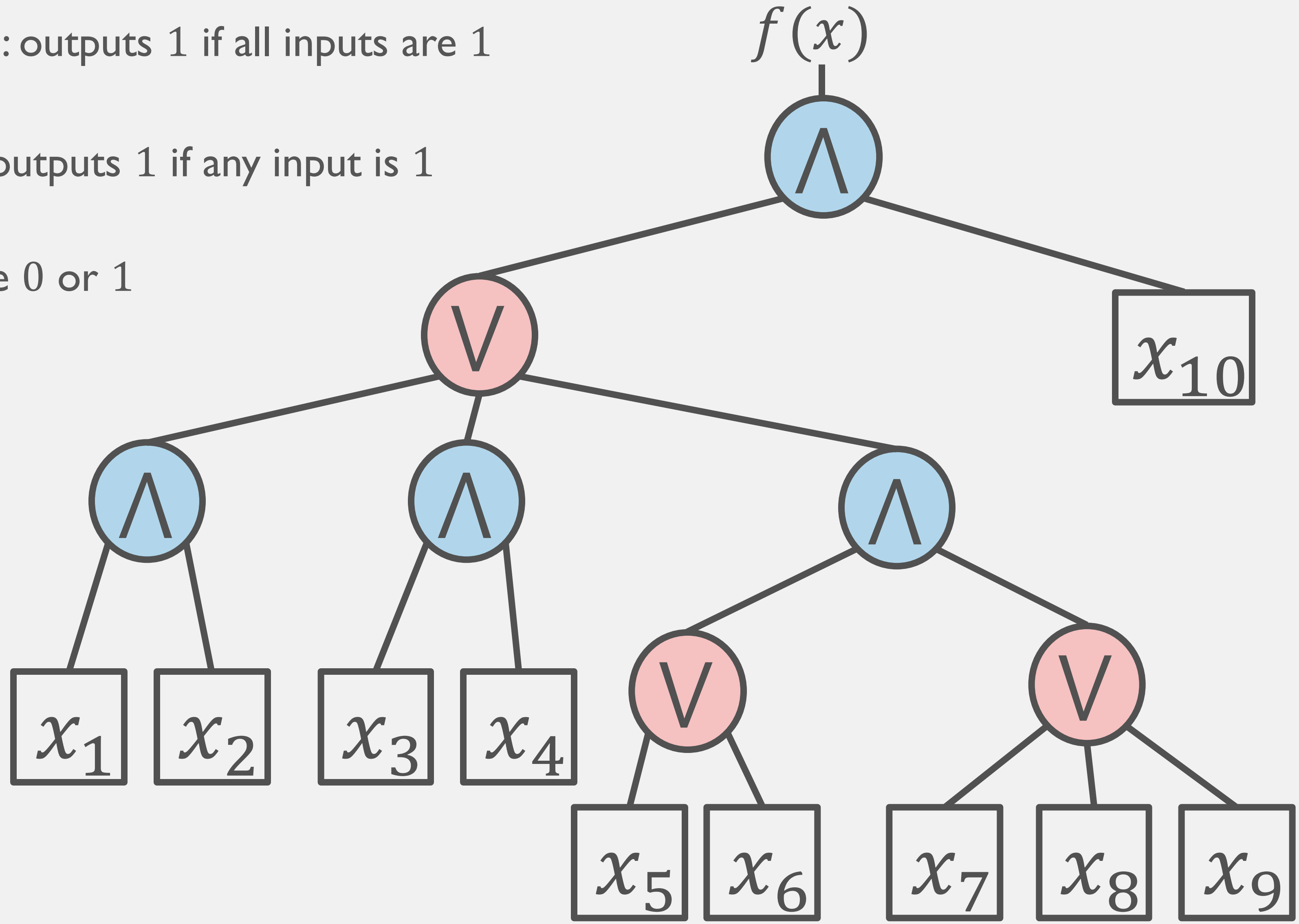
- Statistical inference and machine learning applied to quantum characterization problems
- Quantum complexity theory, especially quantum versions of NP

Boolean Formulas

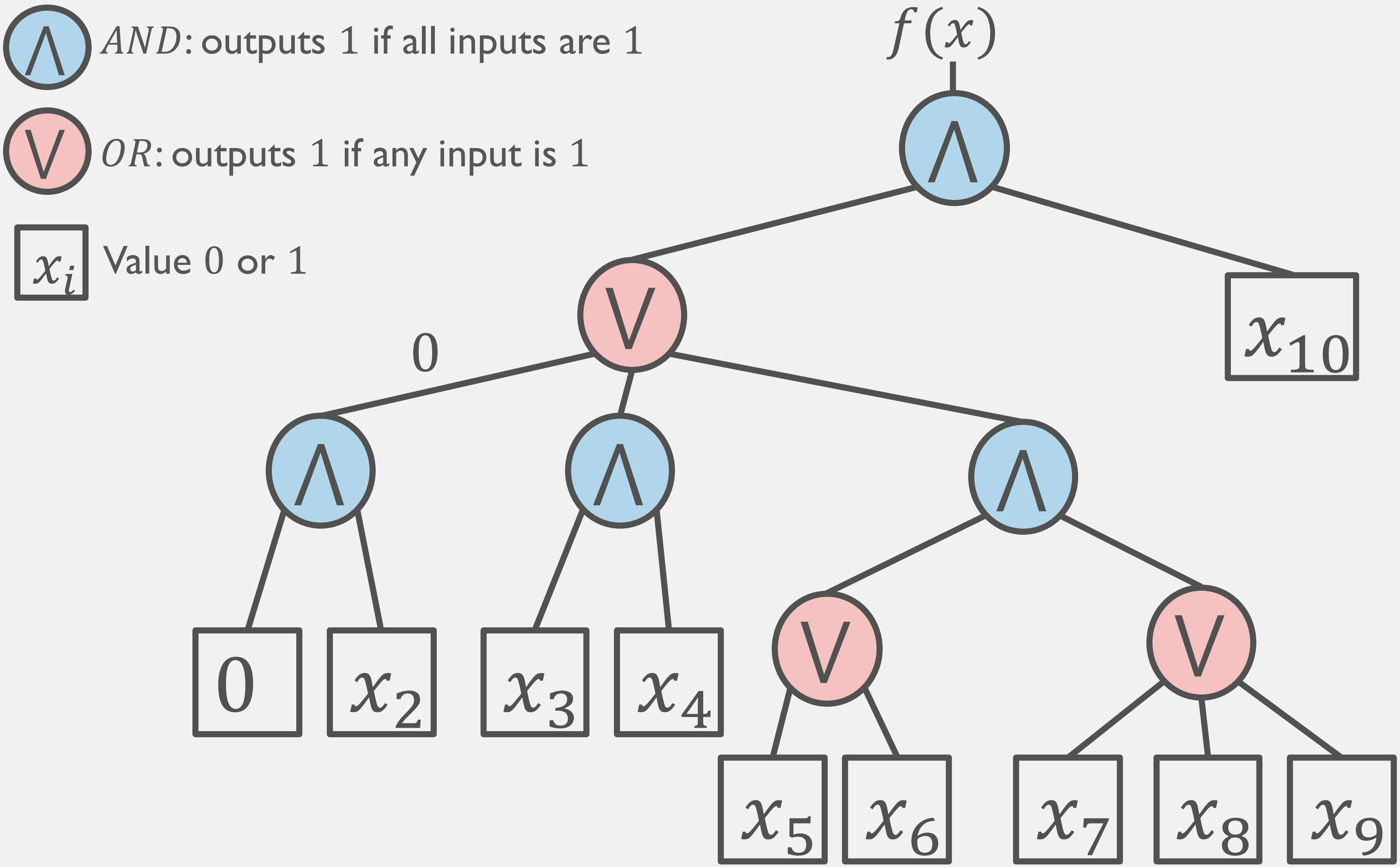
$\wedge$ AND: outputs 1 if all inputs are 1

$\vee$ OR: outputs 1 if any input is 1

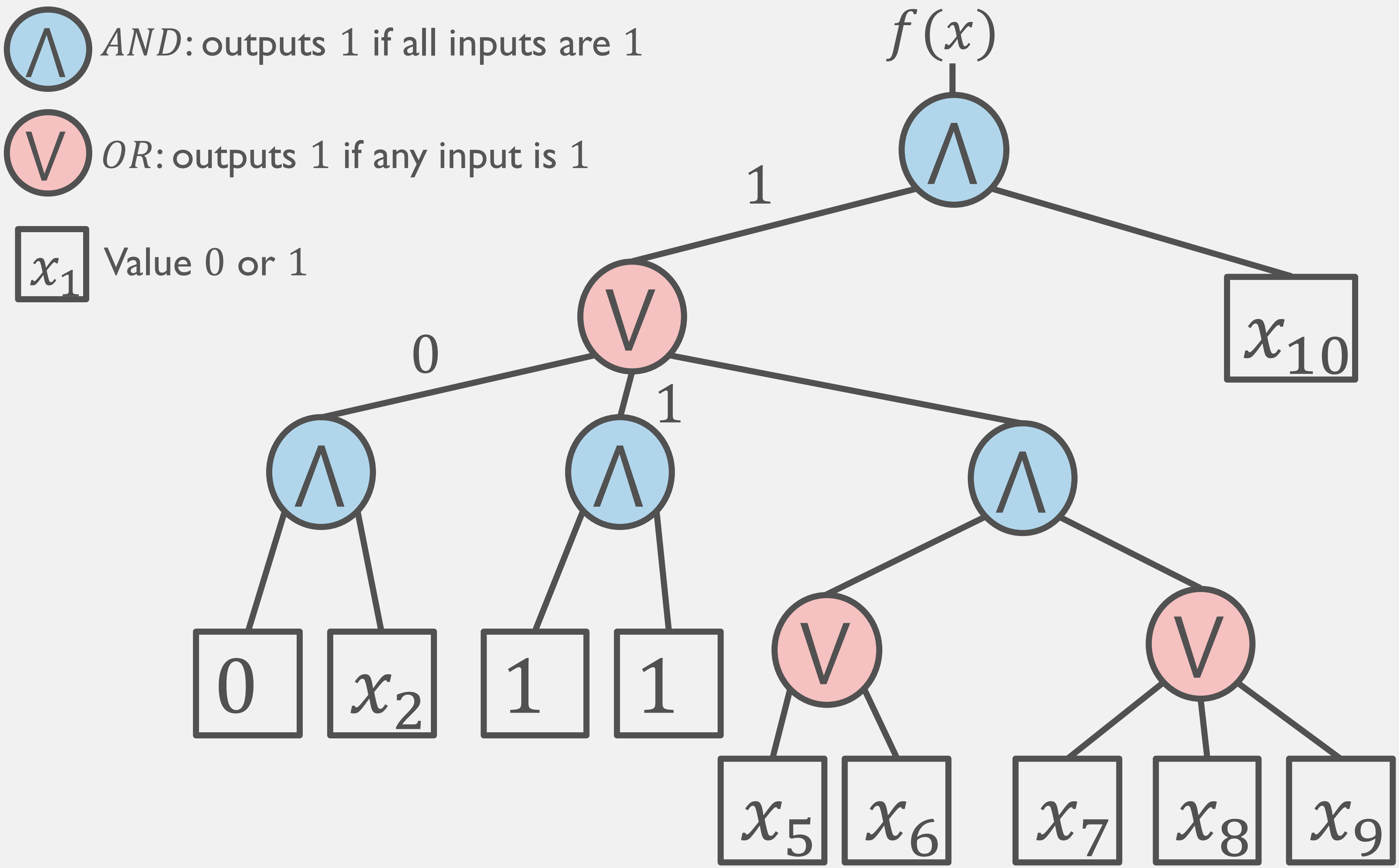
$\boxed{x_i}$ Value 0 or 1



Boolean Formulas



Boolean Formulas



Boolean Formulas

$\wedge$ AND: outputs 1 if all inputs are 1

$\vee$ OR: outputs 1 if any input is 1

$\boxed{x_1}$ Value 0 or 1