Quantum Algorithms for Connectivity: Applications and Analysis

Shelby Kimmel

Middlebury College

Based on work with
Stacey Jeffery: arXiv: 1704.00765 (Quantum vol 1 p 26)
Michael Jarret, Stacey Jeffery, Alvaro Piedrafita, *in progress*

Middlebury

Finding new quantum algorithms

- Quantum computers are almost here!!
- What are we going to do with them?

Finding new quantum algorithms

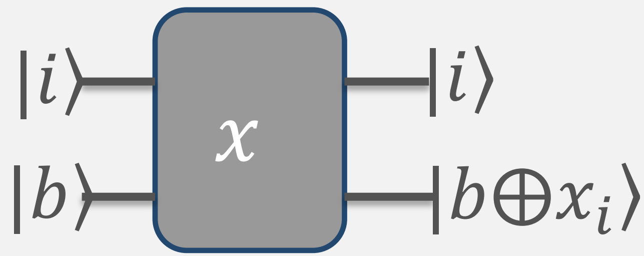
- Find an important problem
  - design a heuristic/provable quantum algorithm
- Better understand existing paradigms
  - Learn what structure quantum algorithms can take advantage of
  - Find problems that fit that structure

Outline

- Quantum Query Algorithms and Span Programs
- Structure of quantum speed-up for st-connectivity
- Applications

Quantum Query Algorithms

Quantum Oracle encodes an n-bit string x:



x_i is ith bit of x

Given an oracle for x, want to evaluate $f(x)$ for $f: \{0,1\}^n \rightarrow \{0,1\}$

Quantum query complexity of f is the number of uses of the oracle needed by a quantum algorithm to evaluate f

Span Programs

$P = (H, U, \tau, A)$

- Finite dimensional inner product spaces $H_1 \oplus \cdots \oplus H_n$, and $\{H_{j,b} \subseteq H_j\}_{b\in\{0,1\}, j\in[n]}$ such that $H_{j,0} + H_{j,1} = H_j$
- Vector space U
- Non-zero target vector $\tau \in U$
- Linear operator $A: H \to U$

Each span program encodes a function $f: \{0,1\}^n \to \{0,1\}$
Each function $f: \{0,1\}^n \to \{0,1\}$ has an infinite number of span program representations

Span Programs

$P = (H, U, \tau, A)$

- Finite dimensional inner product spaces $H_1 \oplus \cdots \oplus H_n$, and $\{H_{j,b} \subseteq H_j\}_{b\in\{0,1\}, j\in[n]}$ such that $H_{j,0} + H_{j,1} = H_j$
- Vector space U
- Non-zero target vector $\tau \in U$
- Linear operator $A: H \to U$

Connection to functions:
$\forall x \in \{0,1\}^n$, let $H(x) = H_{1,x_1} \oplus \cdots \oplus H_{n,x_n}$

$f(x) = 1 \leftrightarrow \exists |w\rangle \in H(x): A|w\rangle = \tau$

Span Programs

$P = (H, U, \tau, A)$

- Finite dimensional inner product spaces $H_1 \oplus \cdots \oplus H_n$, and $\{H_{j,b} \subseteq H_j\}_{b \in \{0,1\}, j \in [n]}$ such that $H_{j,0} + H_{j,1} = H_j$
- Vector space U
- Non-zero target vector $\tau \in U$
- Linear operator $A: H \rightarrow U$

Given a span program for a function f, there is a procedure for creating a query algorithm for f based on that span program, where the query complexity of the algorithm depends only on the span program.

Span Programs

$P = (H, U, \tau, A)$

- Finite dimensional inner product spaces $H_1 \oplus \cdots \oplus H_n$, and $\{H_{j,b} \subseteq H_j\}_{b\in\{0,1\},j\in[n]}$ such that $H_{j,0} + H_{j,1} = H_j$
- Vector space U
- Non-zero target vector $\tau \in U$
- Linear operator $A: H \to U$

There exists a span program that corresponds to a quantum algorithm with optimal query complexity [Reichardt '09,'11]

Span Programs

$P = (H, U, \tau, A)$

- Finite dimensional inner product spaces $H_1 \oplus \cdots \oplus H_n$, and $\{H_{j,b} \subseteq H_j\}_{b \in \{0,1\}, j \in [n]}$ such that $H_{j,0} + H_{j,1} = H_j$
- Vector space U
- Non-zero target vector $\tau \in U$
- Linear operator $A: H \to U$

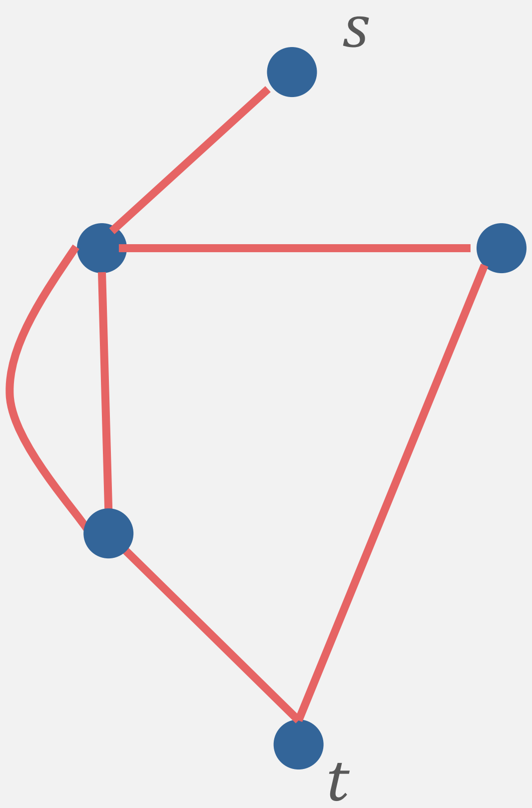
Better understanding of the properties of span programs leads to a better understanding of quantum speed-ups!

Outline

- Quantum Query Algorithms and Span Programs
- Structure of quantum speed-up for st-connectivity
- Applications

st-connectivity

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



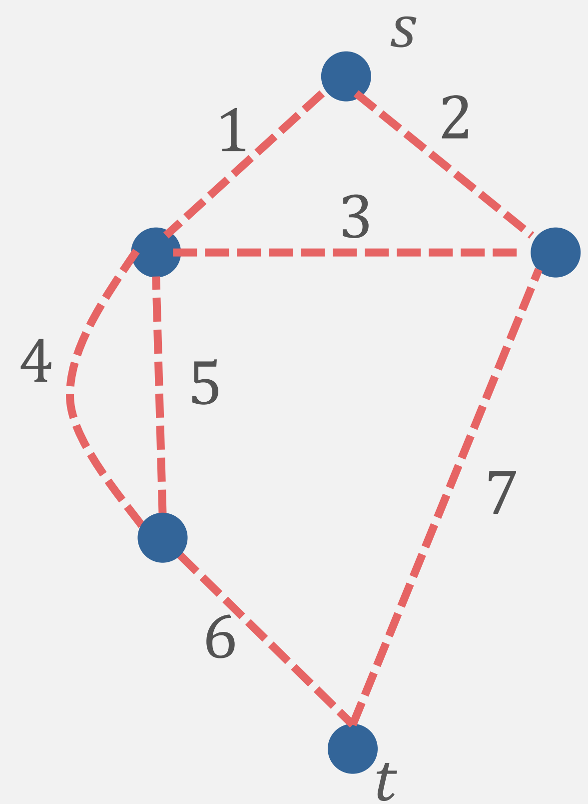
st-connectivity

Is there a path from s to t in a graph G?

$i \rightarrow \boxed{G = (V, E)} \rightarrow e_i$

Edge label

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



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

Span Program for st-connectivity

First described by Karchmer and Wigderson [‘93] when introduced span programs
Used by Belovs and Reichardt [‘12] to create a quantum algorithm for st-connectivity

- $H_{i,1} = \text{span}\{|u, v\rangle, |v, u\rangle : \{v, u\} = edge\ i\}$
- $H_{i,0} = 0$
- $U = \text{span}\{|v\rangle : v \in V\}$
- $A|u, v\rangle = |u\rangle - |v\rangle$
- $\tau = |s\rangle - |t\rangle$

Witness is a path from s to t
Ex: s to u to t: $A(|s, u\rangle + |u, t\rangle) = |s\rangle - |u\rangle + |u\rangle - |t\rangle = |s\rangle - |t\rangle$

Span Program Algorithm for st-connectivity

Nice properties of the span program-based quantum algorithm for st-connectivity

- Uses log-space (candidate for near term devices?) [Belovs & Reichardt '12]
- Under mild assumptions, query complexity equals time complexity [Belovs & Reichardt '12, Jeffery & K '17]
- Now: easier than ever to analyze query complexity

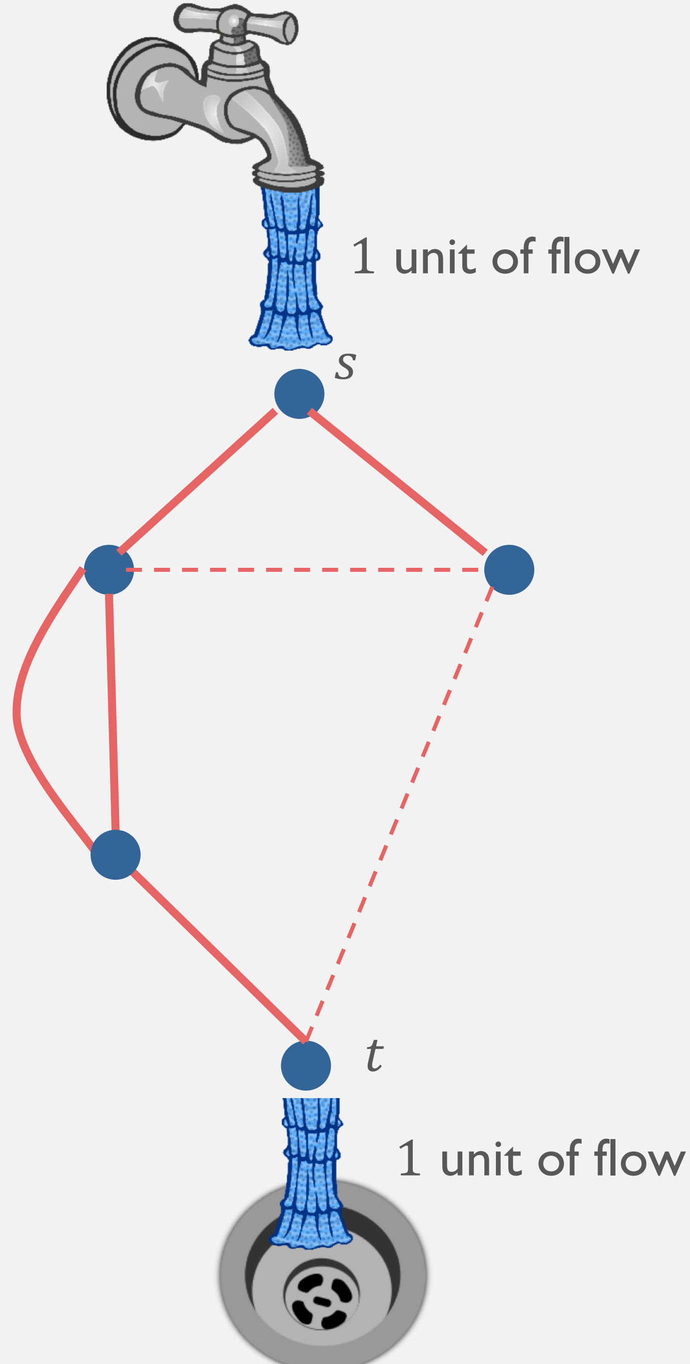
Span Program Algorithm Performance:

Query complexity of span program based st-connectivity algorithm =

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

[Belovs, Reichard, '12] [JJ**K**P, in progress]

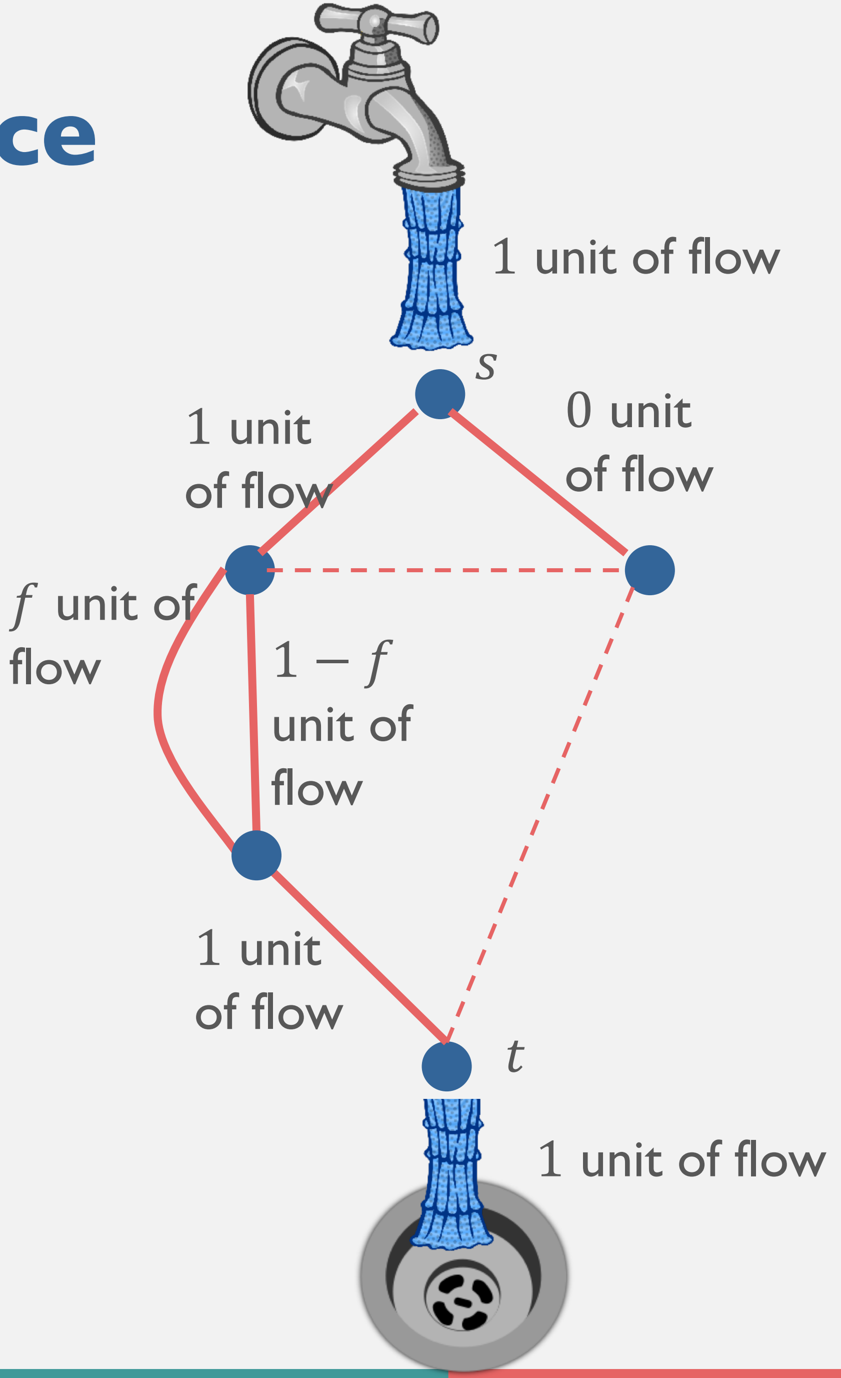
Effective Resistance



Effective Resistance

Valid flow:

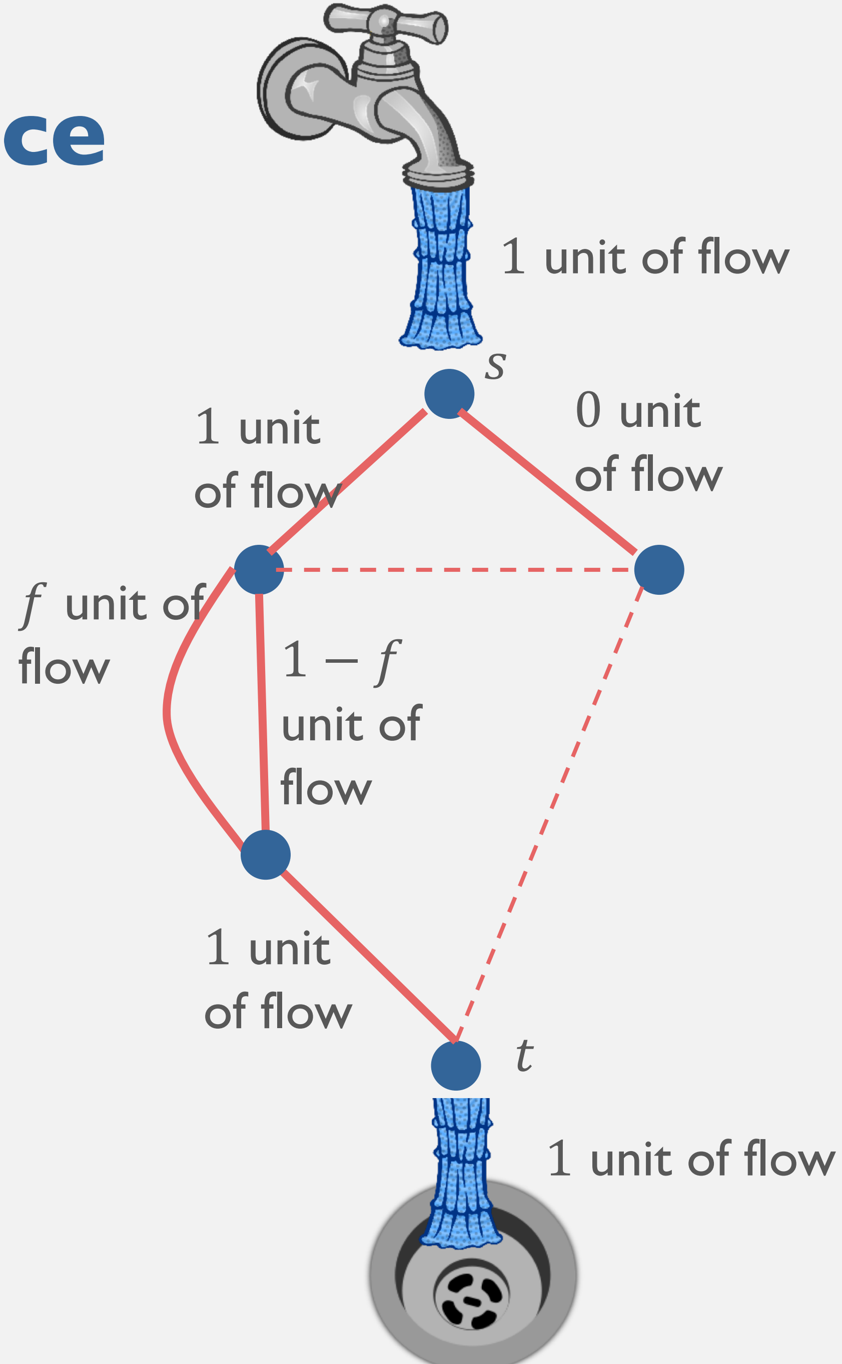
- 1 unit in at s
- 1 unit out at t
- At all other nodes, zero net flow



Effective Resistance

Flow energy:

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

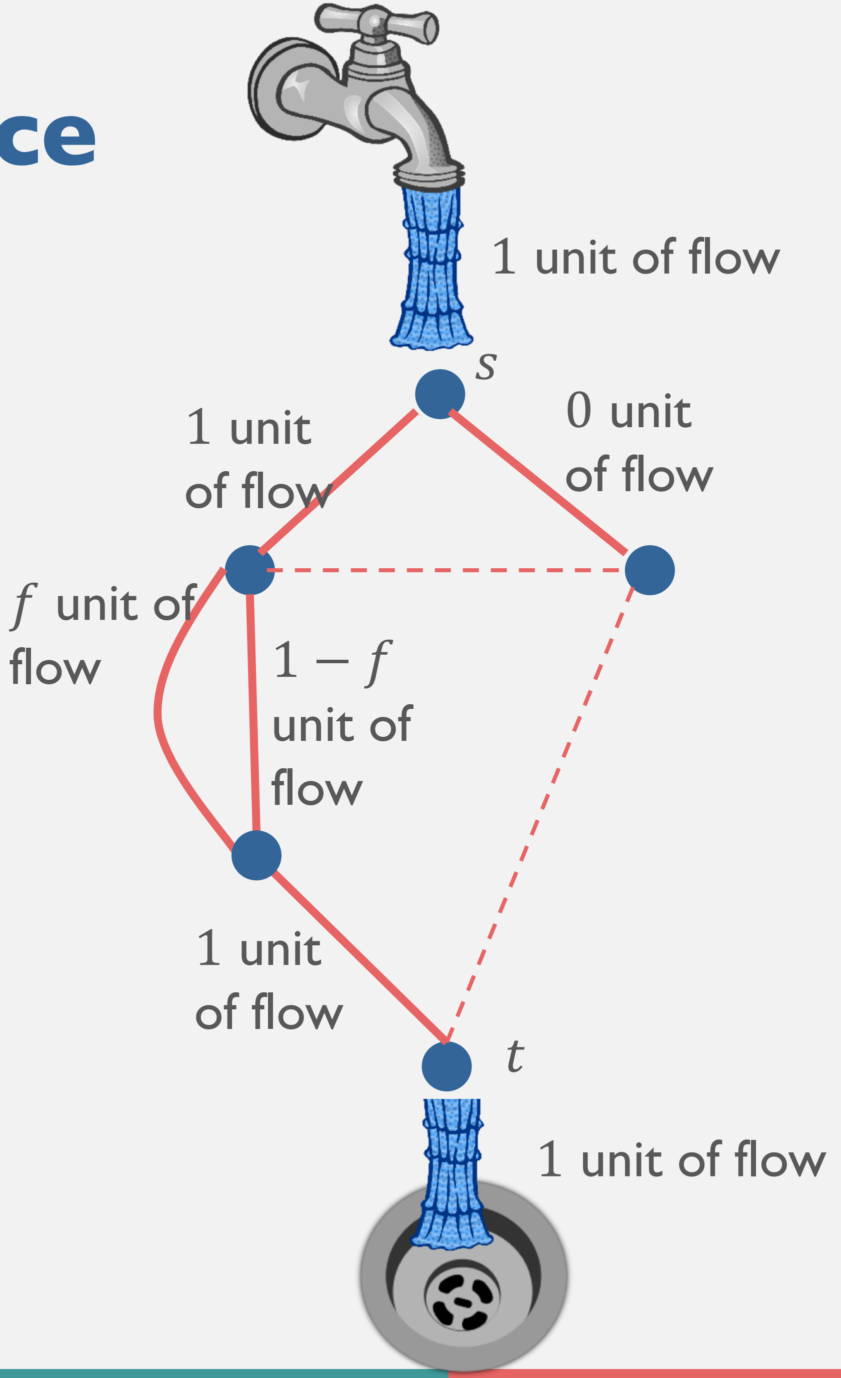


Effective Resistance

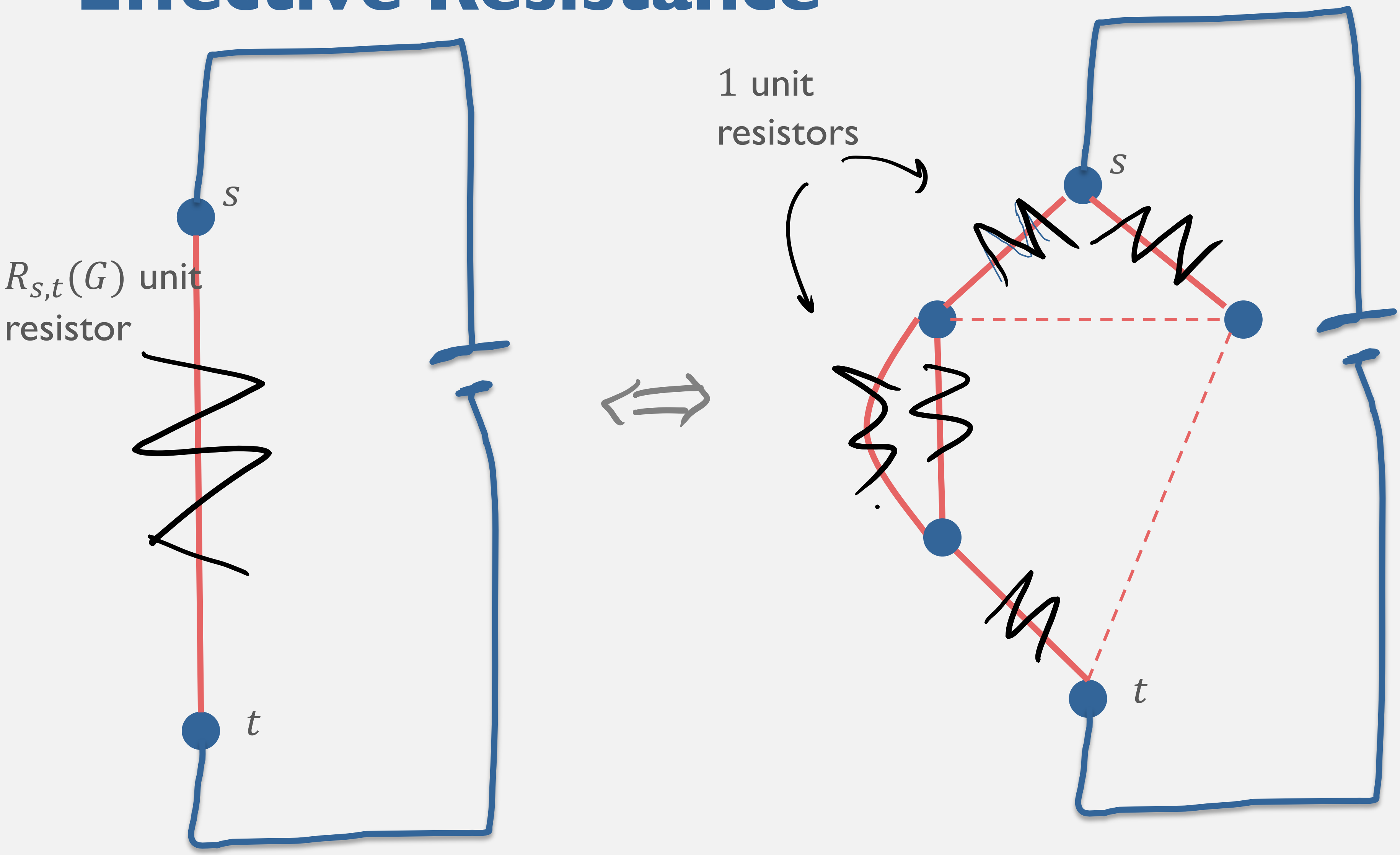
Flow energy:

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

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



Effective Resistance

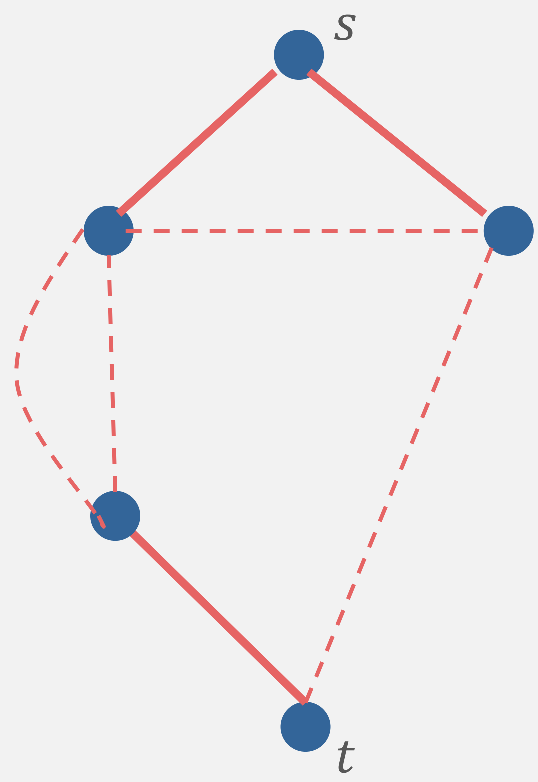


Effective Capacitance

Valid potential energy:

- 1 at s
- 0 at t
- Potential energy difference is 0 across edge

Graph G:

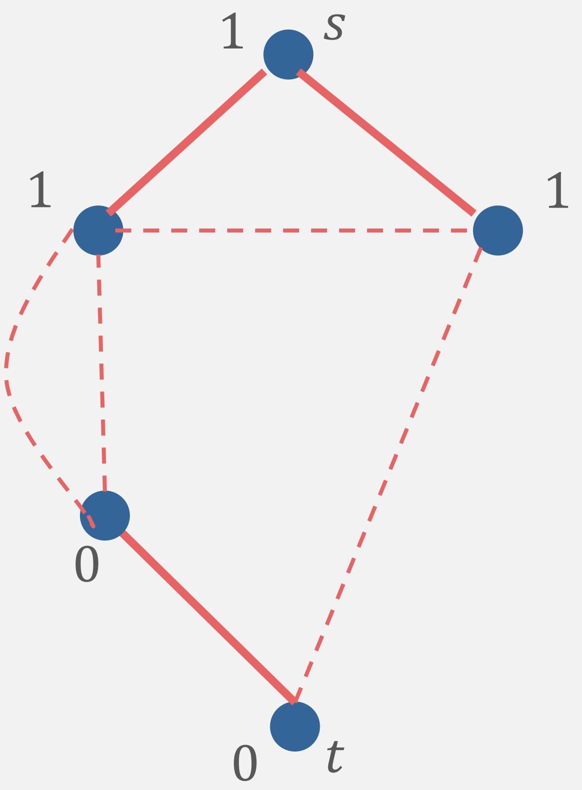


Effective Capacitance

Valid potential:

- 1 at s
- 0 at t
- Potential difference is 0 across edge

Graph G:



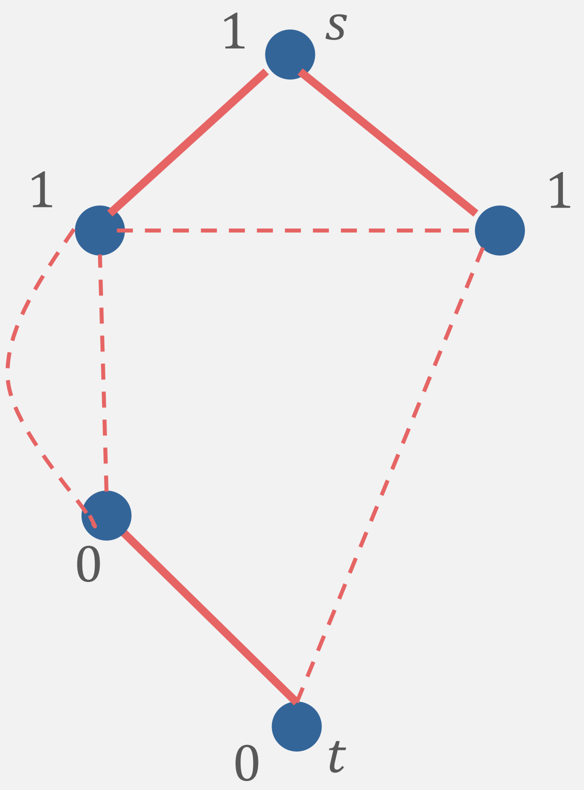
Effective Capacitance

Cut energy:

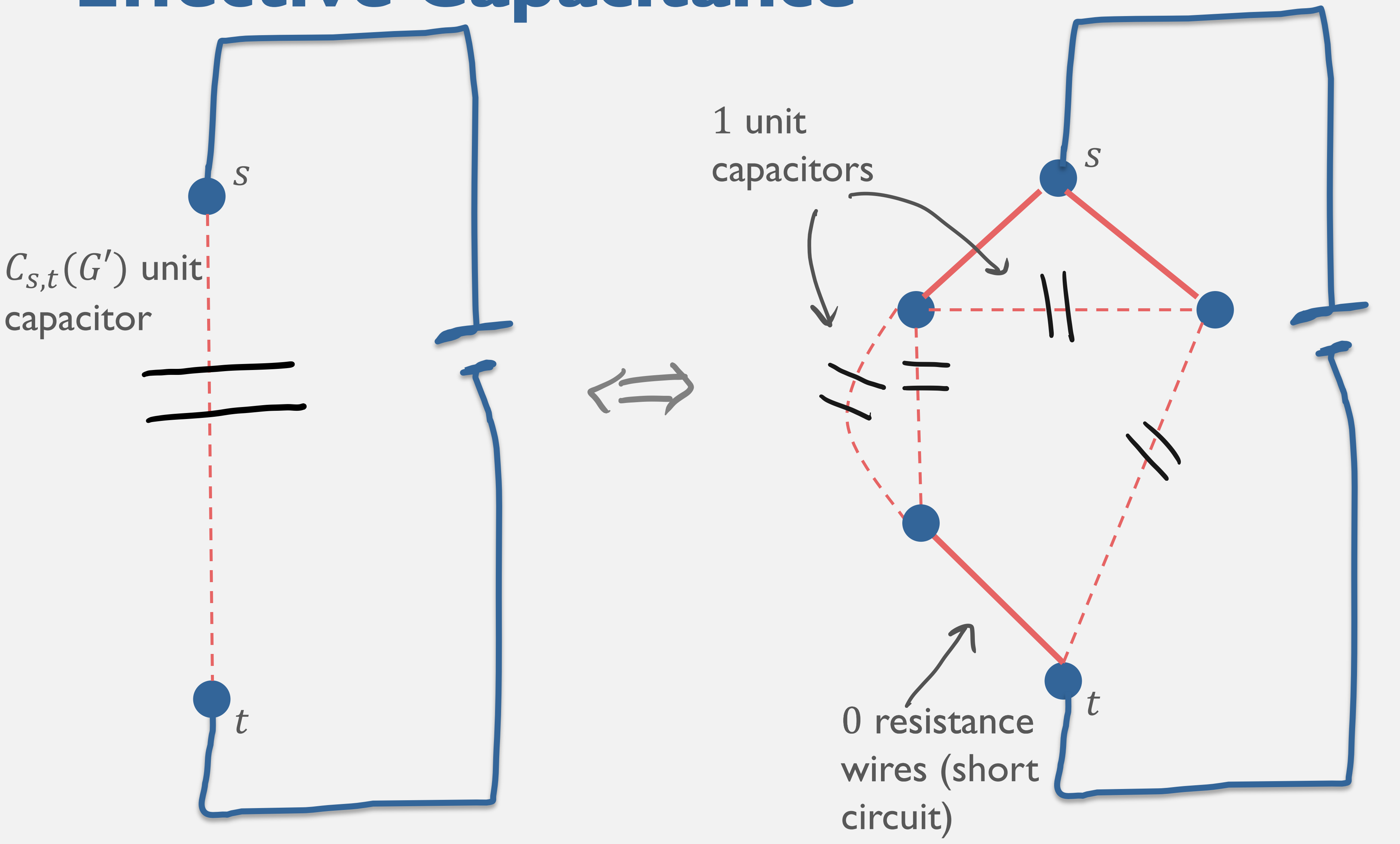
$$\sum_{edges} (Potential\ Difference)^2$$

Effective Capacitance: $C_{s,t}(G)$
Smallest cut energy of any valid potential between s to t on G.

Graph G:



Effective Capacitance



Span Program Algorithm Performance:

Query complexity of span program based st-connectivity algorithm =

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

[Belovs, Reichard, '12]

[JJ**K**P, in progress]

Outline

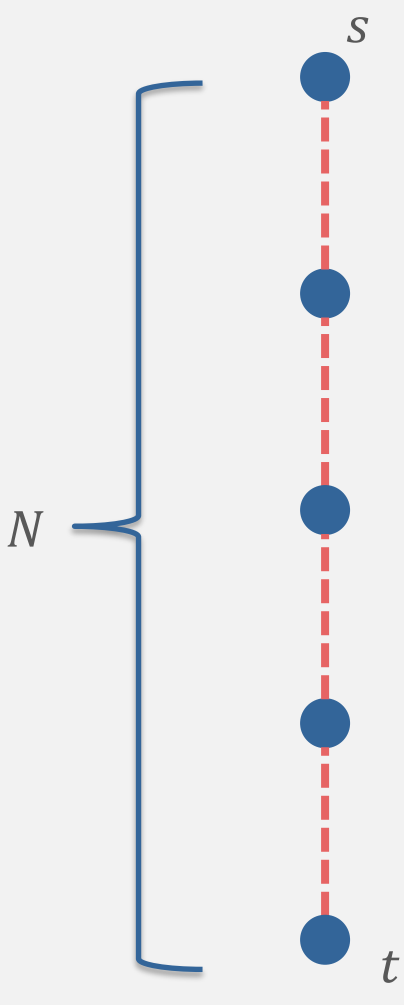
- Quantum Query Algorithms and Span Programs
- Structure of quantum speed-up for st-connectivity
- Applications

Applications: Boolean Formulas

Method of converting Boolean formulas into st-connectivity problems [Nissan and Ta-shma '95, Jeffery & K '17]
Then use quantum algorithm for st-connectivity.

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 if

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

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

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

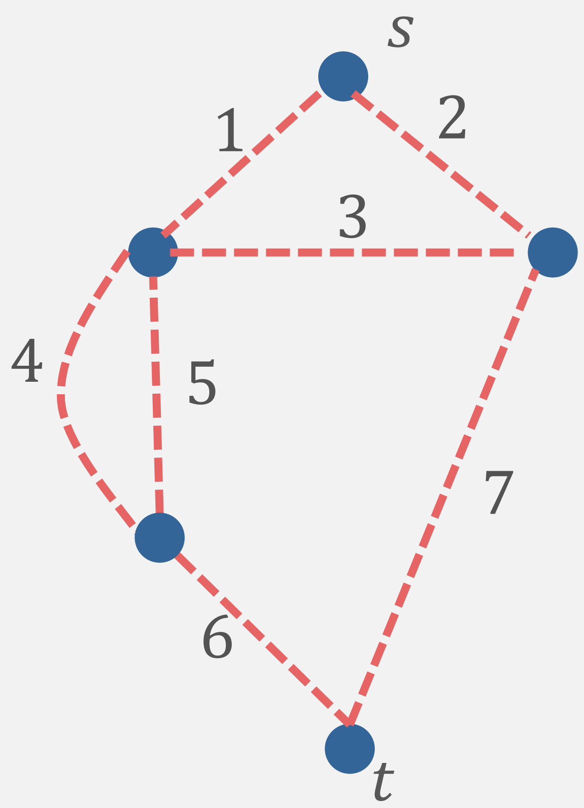
$$N \qquad\qquad 1/\sqrt{N}$$

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

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

Example

Connectivity – is every vertex connected to every other vertex?



Connectivity=
$(st - conn) \wedge (su - conn) \wedge (uv - conn) \ldots$

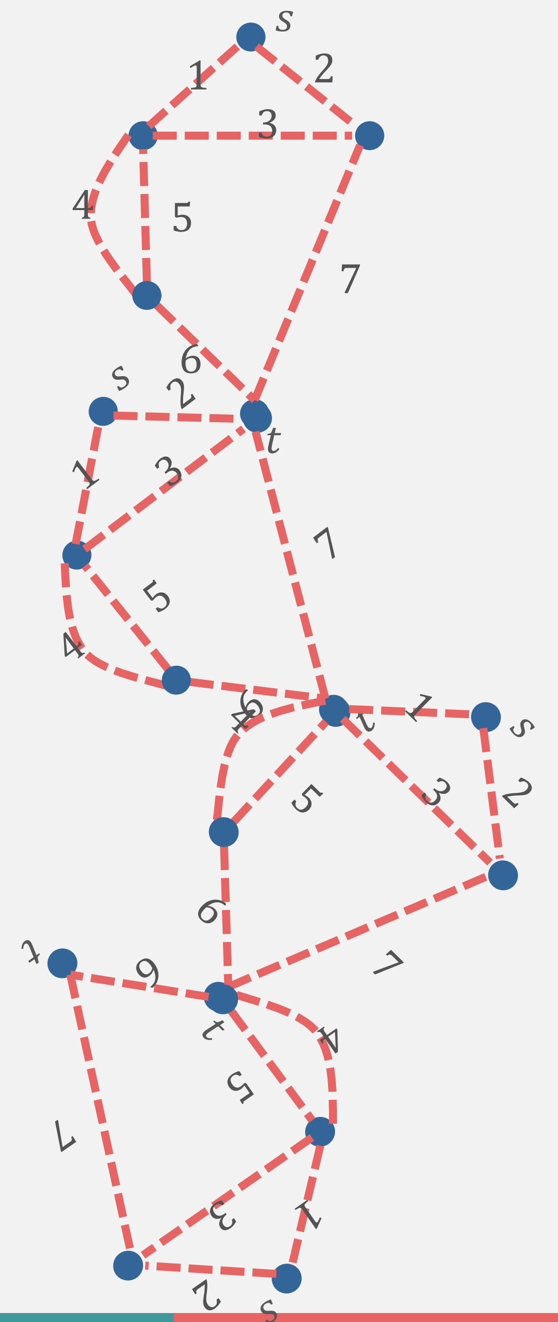
New Example

Connectivity – is every vertex connected to every other vertex?

Results:

- Worst case: $O(V^{3/2})$ (V = # vertices)
- Promised
  - YES – diameter is D
  - NO – κ connected components
  - $O\left(\sqrt{V\kappa D}\right)$

Matches existing algorithms [Durr '06], [Arins '16] for worst case



Open Questions and Current Directions

- There are many other problems than use st-connectivity as a subroutine – does this improved analysis improve the complexity of those algorithms?
- Are there other problems that reduce to st-connectivity?
- What is the classical time/query complexity of st-connectivity in the black box model? Under the promise of small capacitance/resistance?
- Can the effective capacitance/effective resistance analysis be used to understand speed-ups more generally?