Locality & Complexity in Path Search

Andrew Hunter

Senior Thesis

May 4, 2009
Networks are everywhere

- Telephone systems
- Routers

Networks are complicated

- Routing algorithms?
- Link failure?
- Excess traffic?
Purpose

Our goal

Analyze a simplified model of communication.
Channel graph $G$ and free probability $q$
Example
Example
Example
Example
Example
Example
Locality & Complexity in Path Search

Andrew Hunter

Path Search
Motivation
The model
Results on $F_k$

Locality
A (realism?) problem
Our revised model
Locality’s effect
Results

Complexity
Discovering algorithms
Complexity theory crash course
Results
Open questions

Synthesis
Locality’s effect
Open problems
Conclusions
Q&A

Example
Locality & Complexity in Path Search

Andrew Hunter

Path Search
Motivation
The model
Results on $F_k$

Locality
A (realism?) problem
Our revised model
Locality’s effect
Results

Complexity
Discovering algorithms
Complexity theory crash course
Results
Open questions

Synthesis
Locality’s effect
Open problems
Conclusions
Q&A

Example
Example
Example
Example
Locality & Complexity in Path Search

Andrew Hunter

Path Search
Motivation
The model
Results on $F_k$

Locality
A (realism?) problem
Our revised model
Locality’s effect
Results

Complexity
Discovering algorithms
Complexity theory crash course
Results
Open questions

Synthesis
Locality’s effect
Open problems
Conclusions
Q&A

Example
Example
Example
<table>
<thead>
<tr>
<th>Notation &amp; definitions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Channel graph</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>( P_F(G, q) )</td>
</tr>
<tr>
<td>( P_B(G, q) )</td>
</tr>
<tr>
<td>( E(G, q) )</td>
</tr>
</tbody>
</table>

### Channel graph
- **q**: Free probability (edge)
- **p**: blocked probability (edge)
- \( P_F(G, q) \): Free probability (graph)
- \( P_B(G, q) \): blocked probability (graph)
- \( E(G, q) \): Optimal search cost
Previous results

Theorem (Lin & Pippenger)
\[ E(F_k, q) \in \Theta(k) \]

Proof idea.
Previous results

Theorem (Lin & Pippenger)

\[ E(F_k, q) \in \Theta(k) \]

Proof idea.
Previous results

Theorem (Lin & Pippenger)
\[ E(F_k, q) \in \Theta(k) \]

Proof idea.
Previous results

Theorem (Lin & Pippenger)

\[ E(F_k, q) \in \Theta(k) \]

Proof idea.
Previous results

Theorem (Lin & Pippenger)

$E(F_k, q) \in \Theta(k)$

Proof idea.
Previous results

Theorem (Lin & Pippenger)
\[ E(F_k, q) \in \Theta(k) \]

Proof idea.
Previous results

Theorem (Lin & Pippenger)

\[ E(F_k, q) \in \Theta(k) \]

Proof idea.
Previous results

Theorem (Lin & Pippenger)
\[ E(F_k, q) \in \Theta(k) \]

Proof idea.
Previous results

**Theorem (Lin & Pippenger)**

\[ E(F_k, q) \in \Theta(k) \]

**Proof idea.**
Previous results

Theorem (Lin & Pippenger)

\[ E(F_k, q) \in \Theta(k) \]

Proof idea.
Previous results

Theorem (Lin & Pippenger)

\[ E(F_k, q) \in \Theta(k) \]

Proof idea.
Does this match our network model?

If I’m trying to talk to $t$, how do I possibly probe that edge?
What is feasible?
What is feasible?
What is feasible?
What is feasible?
What is feasible?
What is feasible?
What is feasible?
What is feasible?
What is feasible?
What is feasible?
What is feasible?
Local path search

Definition
In *local* path search, we may only probe edges we have a free path to.
How does this affect path search?

Feasibility

- All graphs still searchable
- $P_F(G, q)$

Tractability

- What about $E(G, q)$?
- Seems harder to perform searches
Why is local search harder?

It is much harder to rule out subgraphs.
How bad is DFS now?

Theorem

Searching $F_k$ via DFS (locally) is

$$
\begin{cases}
O(k) & q < \frac{1}{2} \\
O((2q)^k) & \frac{1}{2} < q < \frac{1}{\sqrt{2}} \\
O\left(\frac{1}{q^k}\right) & \frac{1}{\sqrt{2}} < q
\end{cases}
$$

Can we do better?
Lower bounds on $F_k$
Possibly, but not by much.

**Theorem**

For $q > \frac{1}{2}$,\\

$$E(F_k, q) \in \Omega \left( \left( 2q^{-\log q} \right)^k \right).$$
Lower bound for $\frac{1}{2} < q < \frac{1}{\sqrt{2}}$

Proof idea.
Probability of escaping the diverging part of $T_k$ is very low, but a large part of it is accessible.
Lower bound for \( \frac{1}{\sqrt{2}} < q \)

Proof idea.
Complete binary trees are hard to search; the converging tree contains a (relatively deep) one.
Our main result

Proof idea.

Most of the leaves of the converging tree are available; any probe into that tree can only eliminate a few.
What good is an unknown algorithm?

Our mental model

$G$ has some optimal search algorithm $A$ with cost $E(G, q)$. 

Lower bounds are hard!

How do we find that algorithm?

How do we know it is the best?
What good is an unknown algorithm?

Our mental model

$G$ has some optimal search algorithm $A$ with cost $E(G, q)$.

Lower bounds are hard!

How do we find that algorithm?
How do we know it is the best?
Complexity-theoretic path search results

Key observation
Answering many path-search related questions is (probably) very difficult.
Complexity-theoretic path search results

Key observation
Answering many path-search related questions is (probably) very difficult.

![Diagram showing Brute-Force Solution: O(n!), Dynamic Programming Algorithms: O(n²2ⁿ), Selling on eBay: O(1)]
A crash course in complexity theory

Tractable problems

Intractable problems

Locality & Complexity in Path Search
Andrew Hunter

Path Search
Motivation
The model
Results on $F_k$

Locality
A (realism?) problem
Our revised model
Locality’s effect
Results

Complexity
Discovering algorithms
Complexity theory crash course
Results
Open questions

Synthesis
Locality’s effect
Open problems
Conclusions
Q&A
A crash course in complexity theory

Tractable problems
- Graph connectivity

Intractable problems

Or...

\[ s \rightarrow t \]
A crash course in complexity theory

Tractable problems
- Graph connectivity

Intractable problems
- Graph partitioning

Or...

\[ s \rightarrow t \]
A crash course in complexity theory

Tractable problems
- Graph connectivity
- 2CNF satisfiability

Intractable problems
- Graph partitioning

Or...

\[(X \lor Y) \land (Z \lor \neg W) \land \ldots \land (Y \lor \neg Z)\]
A crash course in complexity theory

Tractable problems

- Graph connectivity
- 2CNF satisfiability

Intractable problems

- Graph partitioning
- 3CNF satisfiability

Or…

$$(X \lor Y \lor Z) \land (Z \lor \neg W \lor A) \land \ldots \land (Y \lor \neg Z \lor \neg D)$$
A crash course in complexity theory

Tractable problems

- Graph connectivity
- 2CNF satisfiability
- Linear programming

Intractable problems

- Graph partitioning
- 3CNF satisfiability

Or... 

Maximize 

\[ c^T x \]

subject to 

\[ Ax \leq b \]
A crash course in complexity theory

Tractable problems
- Graph connectivity
- 2CNF satisfiability
- Linear programming

Intractable problems
- Graph partitioning
- 3CNF satisfiability
- Optimal play in Go

Or...
A crash course in complexity theory

<table>
<thead>
<tr>
<th>Tractable problems</th>
<th>Intractable problems</th>
</tr>
</thead>
<tbody>
<tr>
<td>▶ Graph connectivity</td>
<td>▶ Graph partitioning</td>
</tr>
<tr>
<td>▶ 2CNF satisfiability</td>
<td>▶ 3CNF satisfiability</td>
</tr>
<tr>
<td>▶ Linear programming</td>
<td>▶ Optimal play in Go</td>
</tr>
</tbody>
</table>

**Definition**

A *complexity class* is a set of problems all solvable with some restricted set of resources.
The catch

Proving problems intractable is generally out of our reach.

Conjecture \( P \subset NP \).

Conjecture \( NP \subset PSPACE \).
Definition

A problem $X$ is hard for a complexity class $C$ if solving $X$ easily lets us solve anything in $X$ easily. Furthermore, if $X \in C$, we say it is $C$-complete.

The canonical example

Theorem (Cook-Levin)

$3SAT$ is $NP$-complete.
# Classes we’ll need

<table>
<thead>
<tr>
<th>Class</th>
<th>Resources</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P$</td>
<td>Polynomial time</td>
<td>Tractable problems</td>
</tr>
<tr>
<td>$NP$</td>
<td>Polynomial time verifiers</td>
<td>Puzzles</td>
</tr>
<tr>
<td>#$P$</td>
<td>Counting $NP$ solutions</td>
<td>Reliability problems</td>
</tr>
<tr>
<td>$PSPACE$</td>
<td>Polynomial space</td>
<td>Games</td>
</tr>
</tbody>
</table>
Classes we’ll need

<table>
<thead>
<tr>
<th>Class</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P$</td>
<td>Linear programming, shortest path, 3SAT, TSP</td>
</tr>
<tr>
<td>$NP$</td>
<td>Permanent, perfect matchings</td>
</tr>
<tr>
<td>#P</td>
<td>Chess, Go, quantum computing simulations</td>
</tr>
<tr>
<td>PSPACE</td>
<td></td>
</tr>
</tbody>
</table>
Key building block

Theorem (Valiant)

*Calculating* $P_F(G, q)$ *is* $\#P$-complete.

Proof Idea.

- If $q = \frac{1}{2}$, all subgraphs of $G$ are equally likely
- Count connected subgraphs
Cost is at least as hard

**Theorem**

*Calculating $E(G, q)$ is $\#P$-hard.*

**Proof Idea.**
Succinct results

Succinct graphs
Instead of \((V, E)\), we have \((C, n)\) (exponentially smaller)

Theorem

\(\text{Computing } P_F(G, q) \text{ for succinct graphs is } \text{PSPACE-hard}.\)

Proof Idea.
Follows Savitch’s theorem. We can replace connected/disconnected graphs with probably linked/probably blocked graphs.
Succinct results

Theorem

*Computing* $P_F(G, q)$ *for succinct series-parallel graphs is* $PSPACE$-complete.

Proof Idea.

Above graphs can be made series parallel, and probabilities are independent for disjoint parts of SP graphs.
Open questions

Just how hard is search cost?
(We think) a gap exists between \( \#P \) and \( PSPACE \).

How much can we do on succinct graphs?
Only obvious algorithms (except series-parallel) are in \( \text{EXPSPACE} \).
Locality’s effect

Tasks are harder
\[ E(F_k, q) \text{ is exponential} \]

Analyses are easier
We rely on locality in our complexity reductions
Limits of locality

Locality & Complexity in Path Search
Andrew Hunter

Path Search
Motivation
The model
Results on $F_k$

Locality
A (realism?) problem
Our revised model
Locality’s effect
Results

Complexity
Discovering algorithms
Complexity theory crash course
Results
Open questions

Synthesis
Locality’s effect
Open problems
Conclusions
Q&A
Limits of locality
Open problems

Greediness
Is DFS optimal?

Parallel constructions
How can we combine costs from parallel subgraphs?
Complexity problems

Shrinking the upper/lower gap

$\#P \subset PSPACE$ (probably)

Separating probabilities and cost

Is it strictly harder to find algorithms?
Conclusions

Path search
Path search is simple, but rich—still models many of the facets of networks

Locality
Restricting our model has made it considerably more interesting.
Questions?