Securely Solving Simple Combinatorial Graph Problems

Abdelrahaman Aly   Edouard Cuvelier   Sophie Mawet
Olivier Pereira   Mathieu Van Vyve

4 April 2013

Motivation

We investigate the problem of securely solving graph problems:

- in a multi-party setting,
- when the knowledge of the graph is distributed.

Example of applications include:

- privacy-preserving GPS guidance,
- privacy-preserving determination of topological features in social networks,
- privacy-preserving benchmarks between competing network operators.

Contributions

New protocols for securely solving graph problems.

- The shortest path problem:

	Original	Secret weights	Secret structure
Bellman-Ford	$\|V\|\|E\|$	$\|V\|\|E\|$	$\|V\|^3$
Dijkstra	$\|V\|^2$	$\|V\|^3$	$\|V\|^3$

- The maximum flow problem:

	Original	Secret weights	Secret structure
Edmonds-Karp	$\|V\|\|E\|^2$	$\|V\|\|E\|^2$	$\|V\|^5$
Push-Relabel	$\|V\|^3$	$\|V\|^2\|E\|$	$\|V\|^4$

Challenges

Challenges related to securely solving graph problems.

- **Leakage by execution flow**: running time, memory addressing, ... usually depend on the data that are manipulated.
- **Different efficiency metrics**: The traditional complexity metrics do not transpose to secure computations.
- **Composability**: The algorithm should leak no partial solution.

Challenge 1

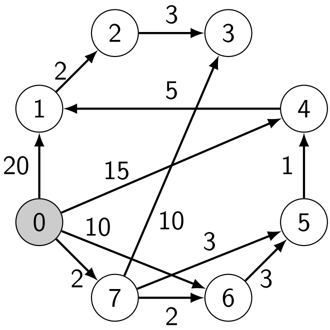
Leakage by execution flow: running time, memory addressing, . . .
usually depend on the data that are manipulated.

Challenge 1

Leakage by execution flow: running time, memory addressing, ... usually depend on the data that are manipulated.

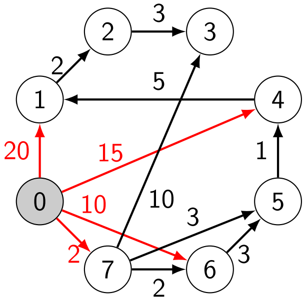
Dijkstra's algorithm maintains for each vertex:

- the status (unreached, labelled, scanned),
- the current previous vertex,
- the current distance.

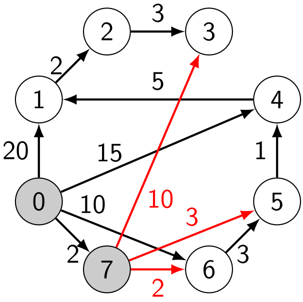


Leakage by execution flow

Dijkstra's first iteration:

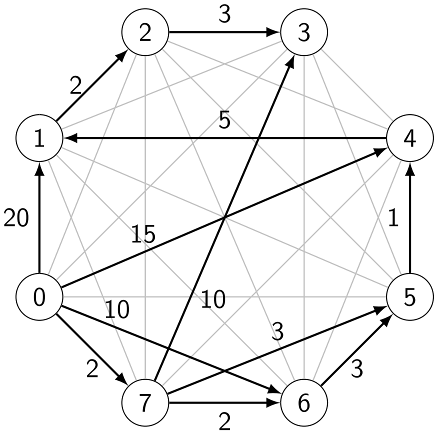


Dijkstra's second iteration:



We need to hide the scanning sequence.

We consider a complete graph to preserve privacy!



Challenge 2

Different efficiency metrics: The traditional complexity metrics do not transpose to secure computations.

One comparison costs more than 100 multiplications.

Challenge 2

Different efficiency metrics: The traditional complexity metrics do not transpose to secure computations.

One comparison costs more than 100 multiplications.

Complexity for a graph with V vertices and E edges:

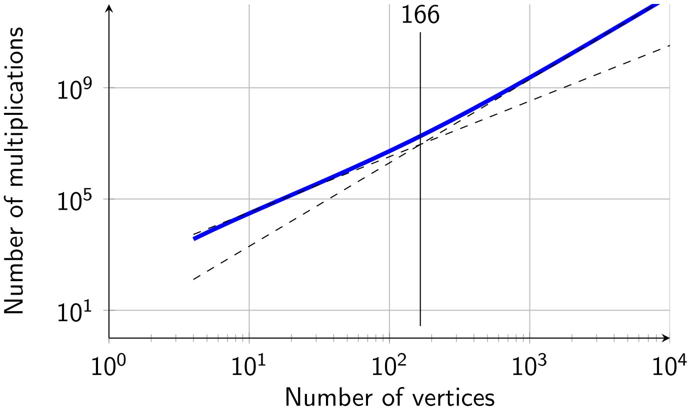
Dijkstra's complexity:

- $O(V^2)$ comparisons
- $O(V^3)$ multiplications

Bellman-Ford's complexity:

- $O(V \cdot E)$ comparisons
- $O(V \cdot E)$ multiplications

Number of multiplications for Dijkstra's algorithm



The dashed lines highlight the quadratic then cubic growths.

Challenge 3

Composability: The algorithm should leak no partial solution.

Challenge 3

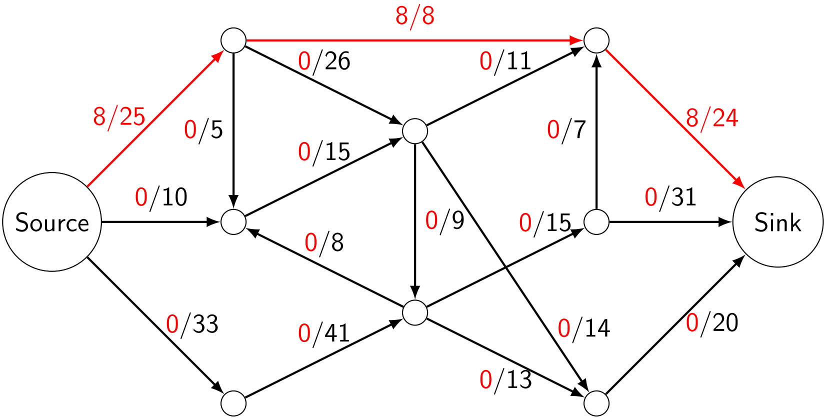
Composability: The algorithm should leak no partial solution.

The maximum flow algorithm makes use of the secure shortest path (which cannot leak any partial information).

Brickell and Shmatikov proposed a shortest path solution that revealed a part of the solution at each step. [BS05]

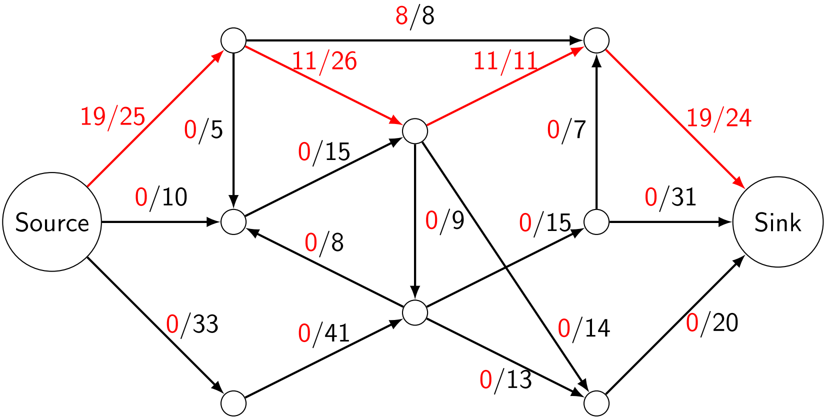
Edmonds-Karp's algorithm

Find the smallest augmenting path in the residual graph in $O(E)$



Edmonds-Karp's algorithm

Find the smallest augmenting path in the residual graph in $O(E)$



Number of steps is at most E, length of path is at most $V - 1$

Secure Maximum Flow based on Edmonds-Karp

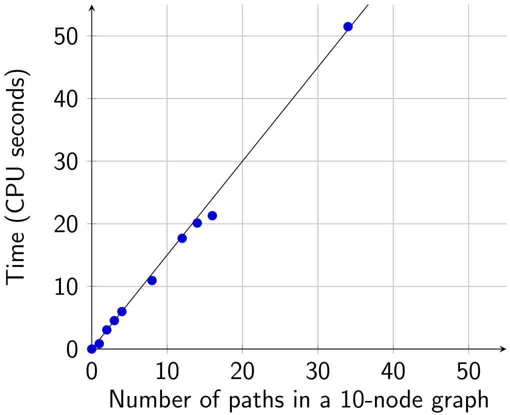
- dynamic search of the smallest augmenting path is tricky
- hide the length of the paths
- keep the time of execution reasonable

Secure solution for the Maximum Flow

Consider all the paths (sorted) *even if they are not augmenting!*

- ~~dynamic search of the smallest augmenting path is tricky~~
- ~~hide the length of the paths~~
- keep the time of execution reasonable

Results for the secure Maximum Flow



The number of paths has to be small: $< E^2$

Conclusion

Our investigation raised interesting complexity gaps between centralized algorithms and secure protocols.

Further work:

- Design efficient datastructures (for example priority queues [Toft12]),
- Trade secure comparisons for cheaper arithmetic operations.

Thank you for your attention!