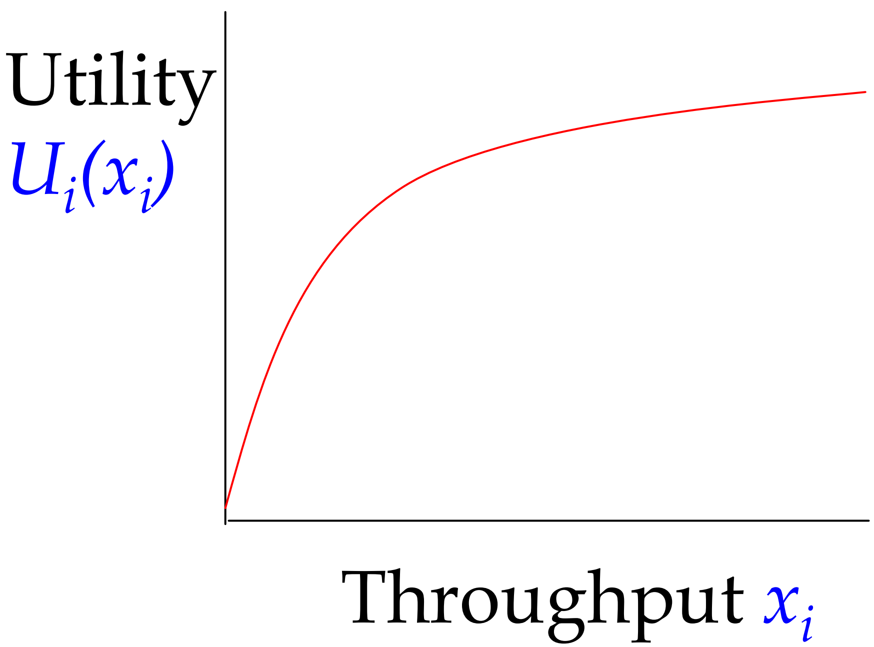
Congestion Control Model

Users are indexed by i



aggregate utility

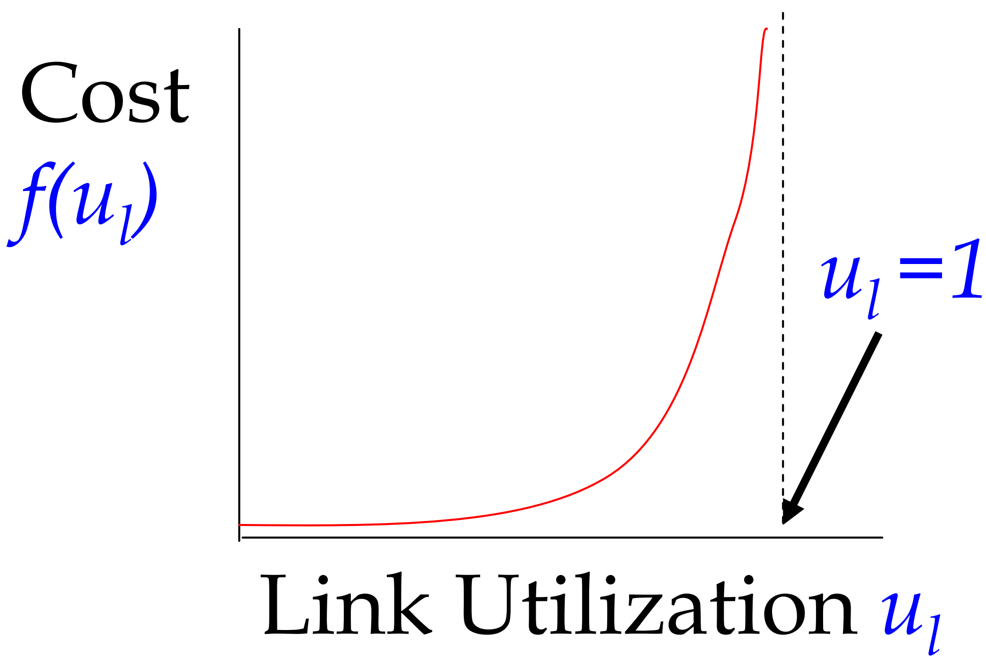
$$\text{max.} \sum_i U_i(x_i)$$
$$\text{s.t.} \quad Rx \le c$$

capacity constraints

Congestion control provides fair rate allocation amongst users

Traffic Engineering Model

Links are indexed by l



$$\text{min. } \sum_l f(u_l)$$

aggregate cost

Traffic engineering avoids bottlenecks in the network

Motivation

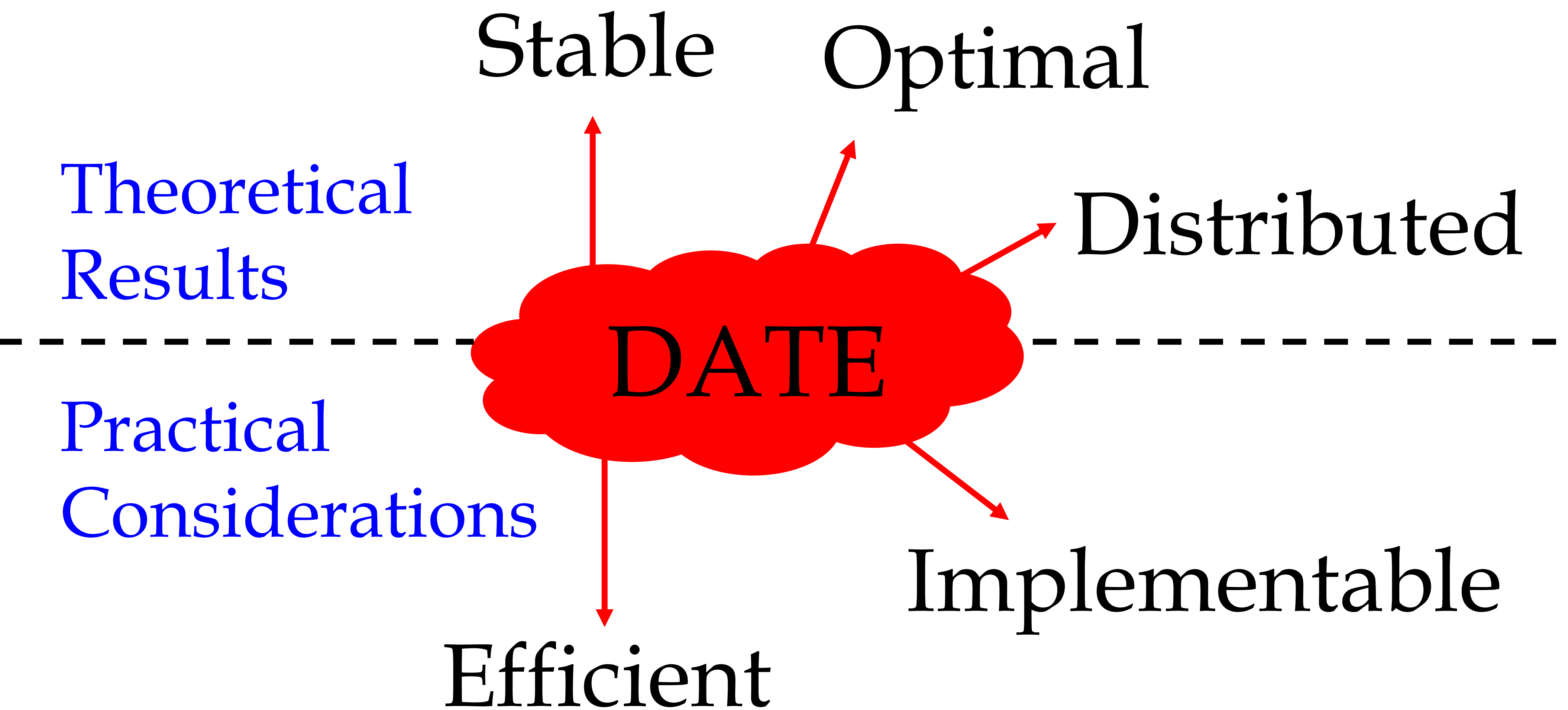
Congestion
Control
User
Performance

Traffic
Engineering
Network
Robustness

Can be at odds!

To balance performance and robustness, we chose max. $\sum_i U_i(x_i) - \sum_l f(u_l)$ as our objective.

Design Goals



Theoretical Results

- Theorem 1: The DATE algorithm converges to the optimum of max. $\sum_i U_i(x_i) - \sum_l f(u_l)$ for sufficiently small step sizes.

DATE:
Dual driven algorithm
Users choose throughput

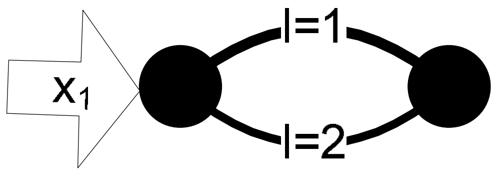
Theoretical Difference

Other Work:
Primal driven algorithm
Users choose prices

Practical Difference

Achieving Stability

- Distributed routing can be unstable.



If you initially route on the top path, then the bottom path is not loaded, causing oscillations.

- Problem: No coordination between measured link load and target link load.
- We introduce consistency price to perform the coordination.

Implementation Challenges

Router Hardware:

- Per flow policing
- Edge routers need to split traffic

Router Software:

- Establishing multiple paths between edge routers
- Frequent link utilization feedback
- Added computation at routers

Conclusions

- DATE balances performance for users and robustness for the network.
- Theoretical analysis shows DATE is stable, optimal and distributed.
- Ongoing simulations will test implementability and efficiency.
- Can explore an architecture where only long-lived flows are routed using DATE.