
ORACLE®

ORACLE

Constrained Data-Driven Parallelism

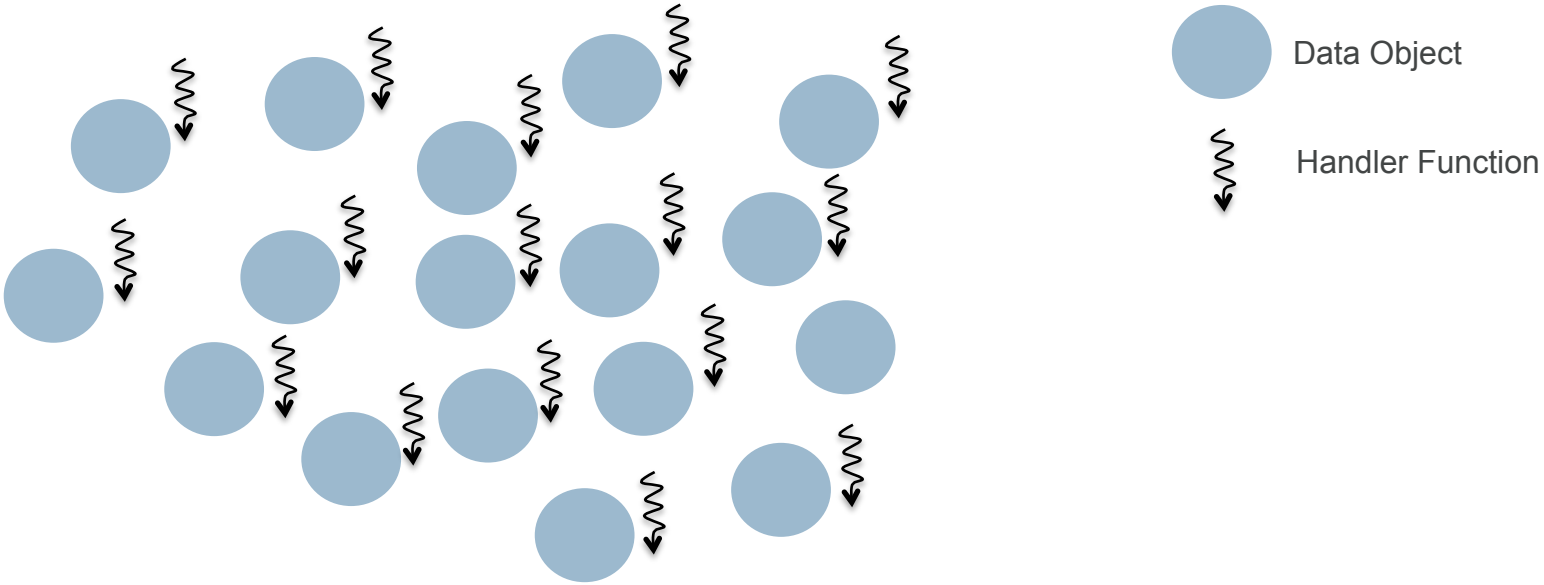
Virendra J. Marathe

with
Tim Harris
Yossi Lev
Victor Luchangco
Mark Moir

Scalable Synchronization Research Group
Oracle Labs

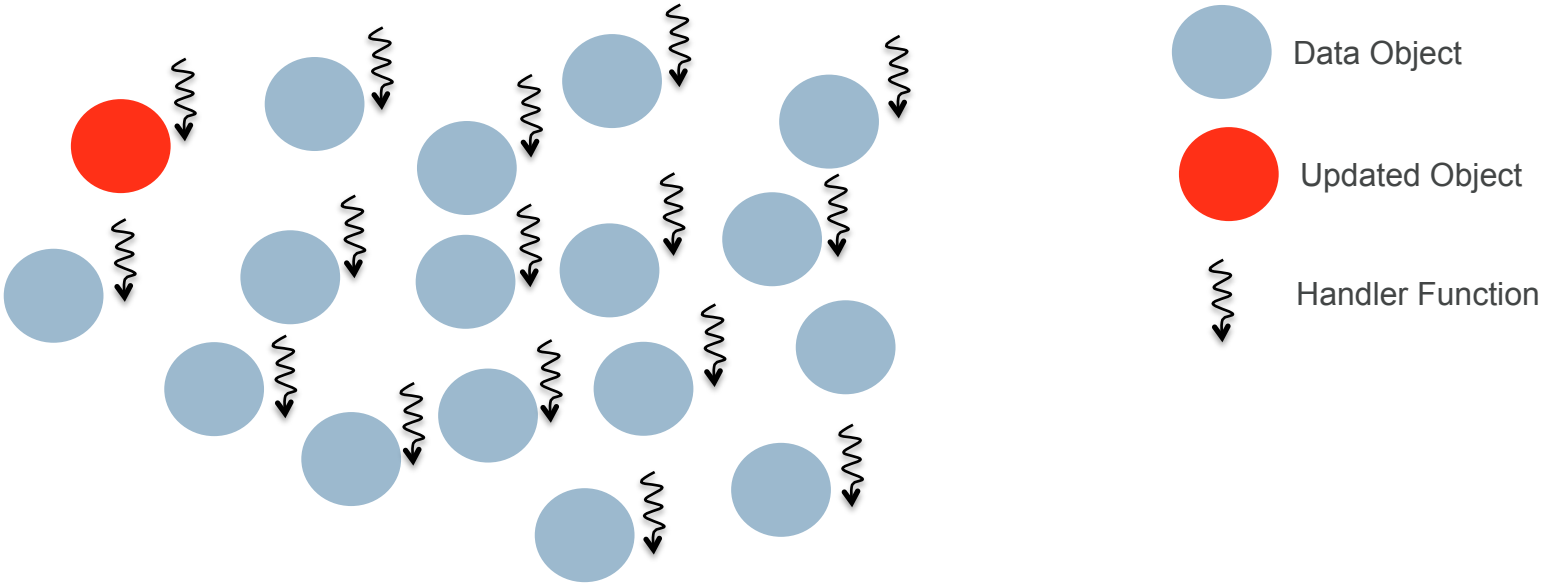
Data-Driven Parallelism

Parallel computation driven by data updates



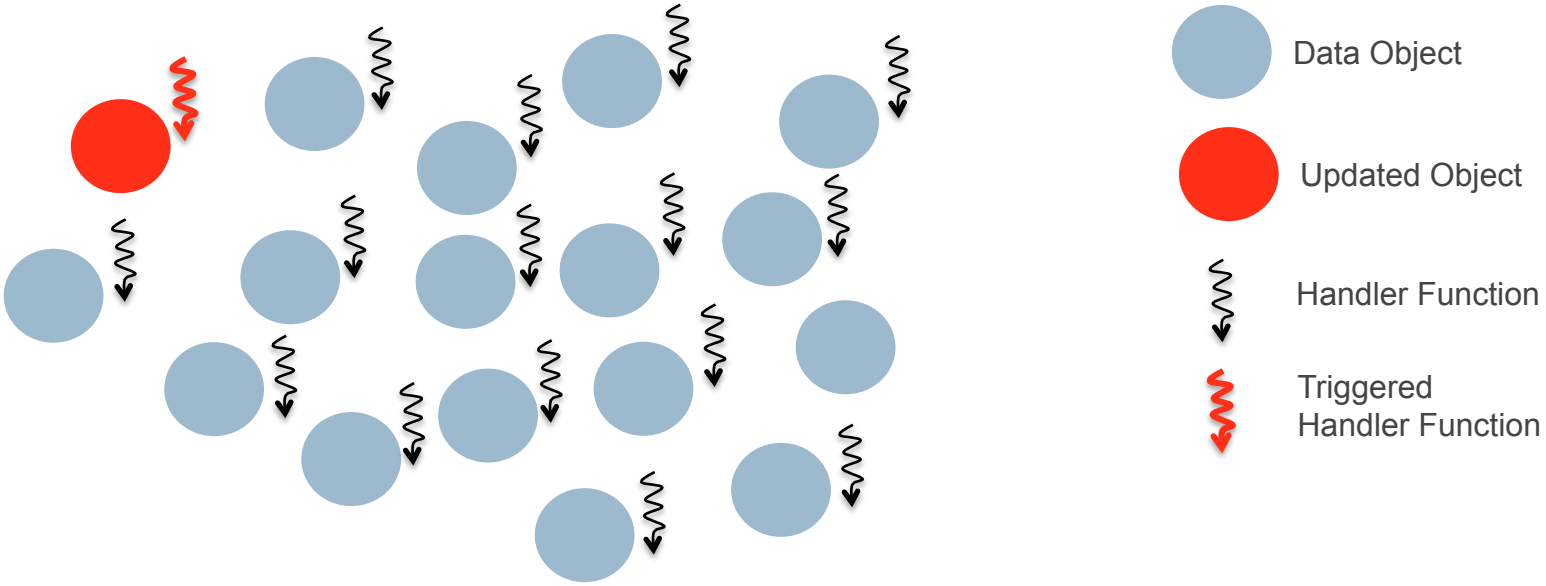
Data-Driven Parallelism

Parallel computation driven by data updates



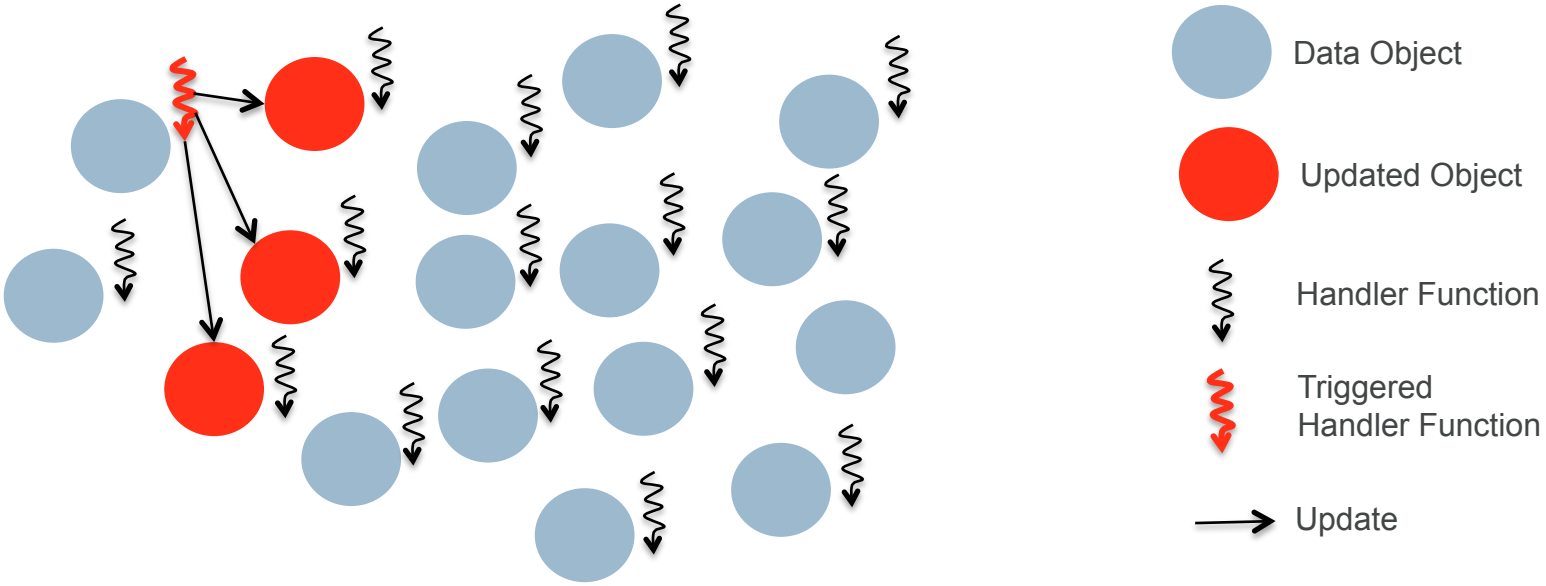
Data-Driven Parallelism

Parallel computation driven by data updates



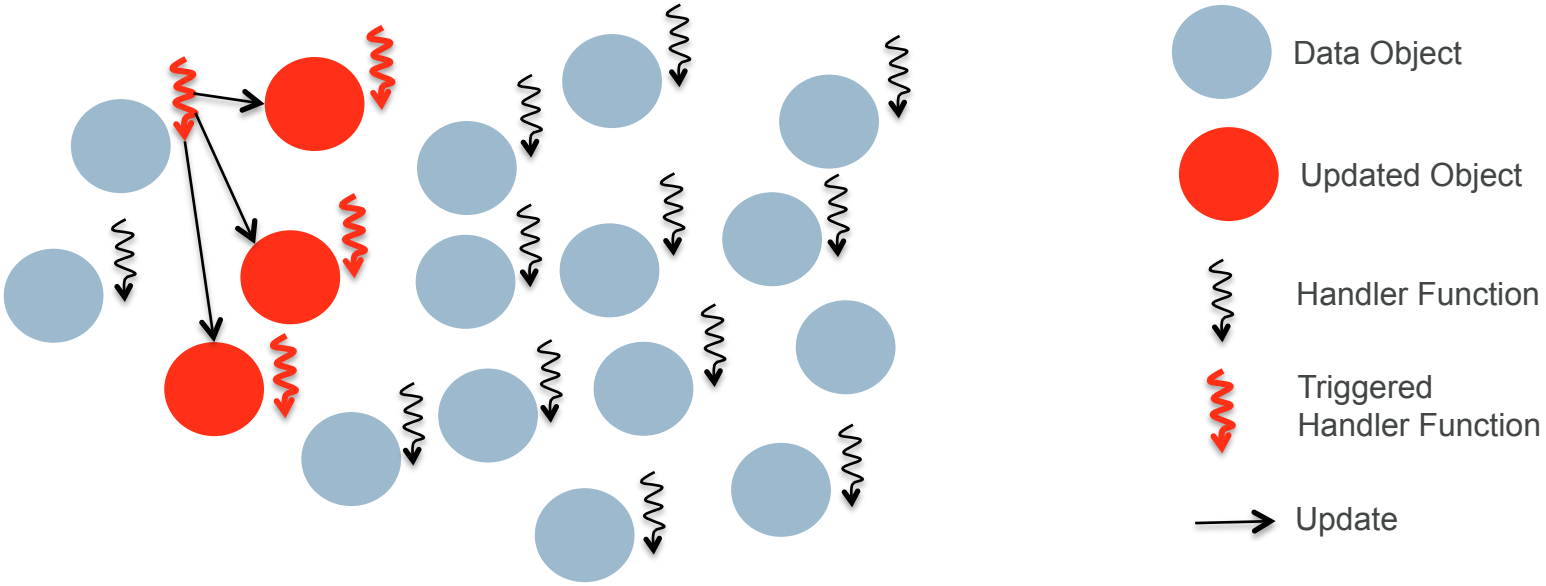
Data-Driven Parallelism

Parallel computation driven by data updates



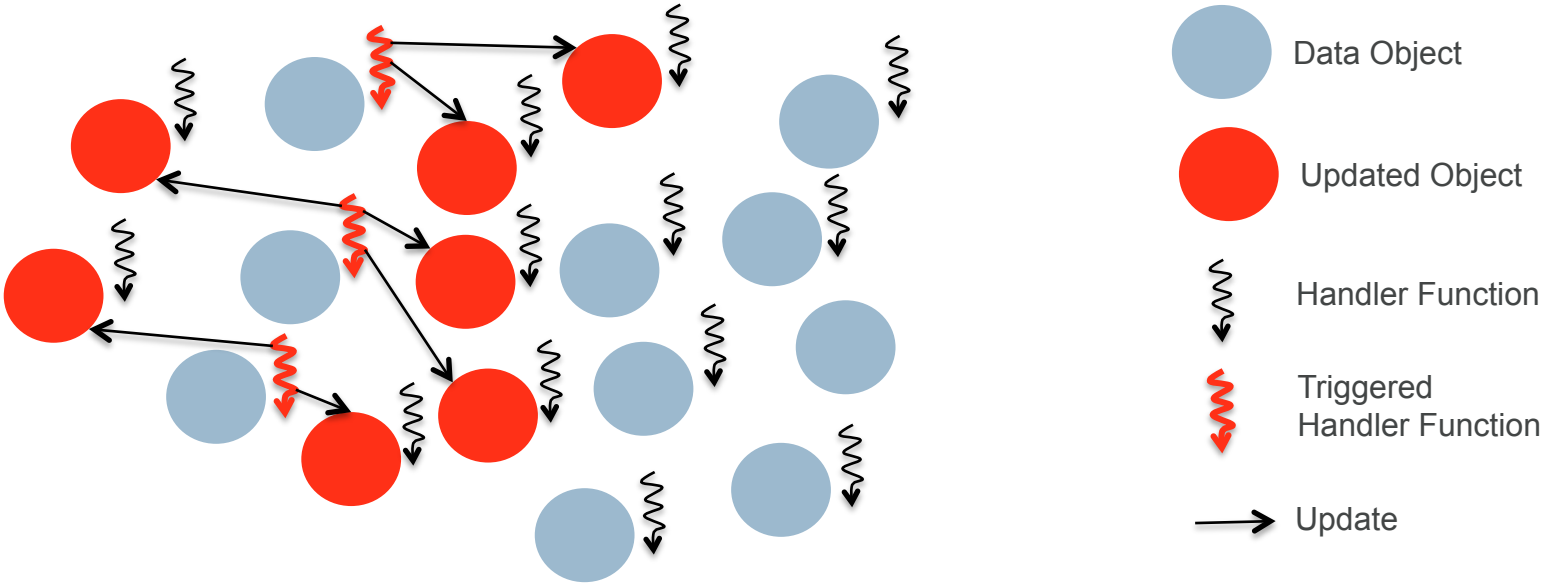
Data-Driven Parallelism

Parallel computation driven by data updates



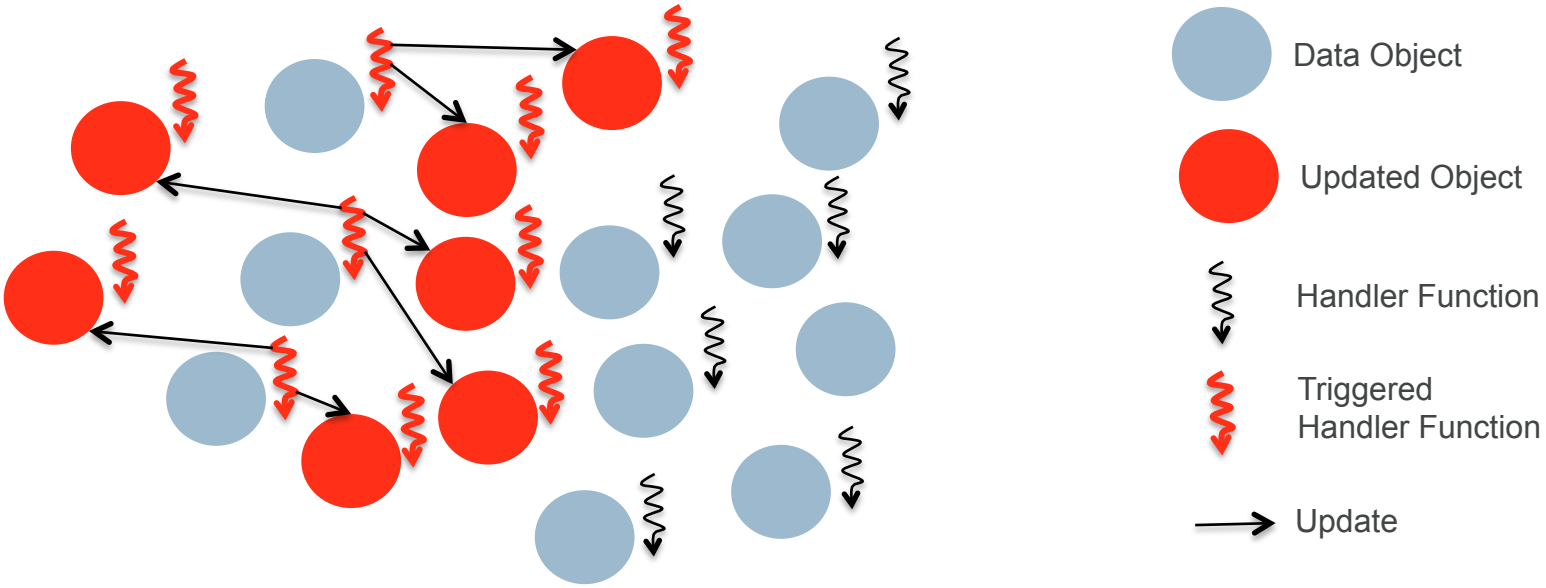
Data-Driven Parallelism

Parallel computation driven by data updates



Data-Driven Parallelism

Parallel computation driven by data updates



Data-Driven Parallelism

Great, but not good enough!

Data-Driven Parallelism

Great, but not good enough!

Does not apply to lots of problem classes

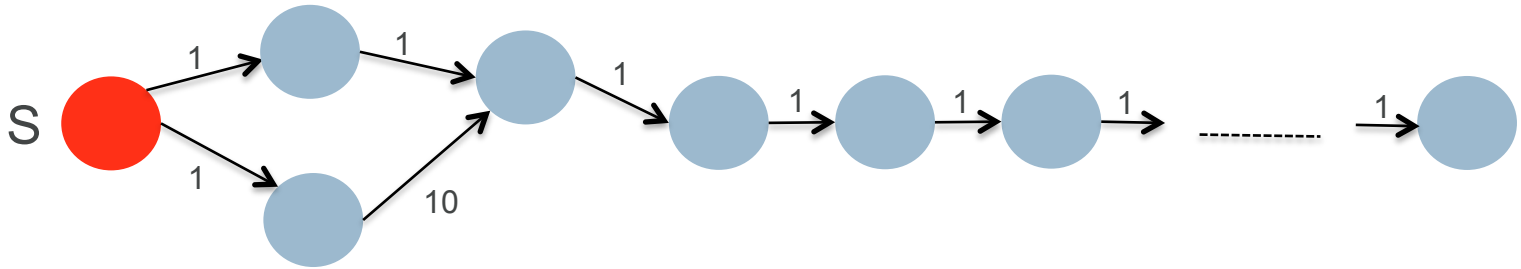
Data-Driven Parallelism

Great, but not good enough!

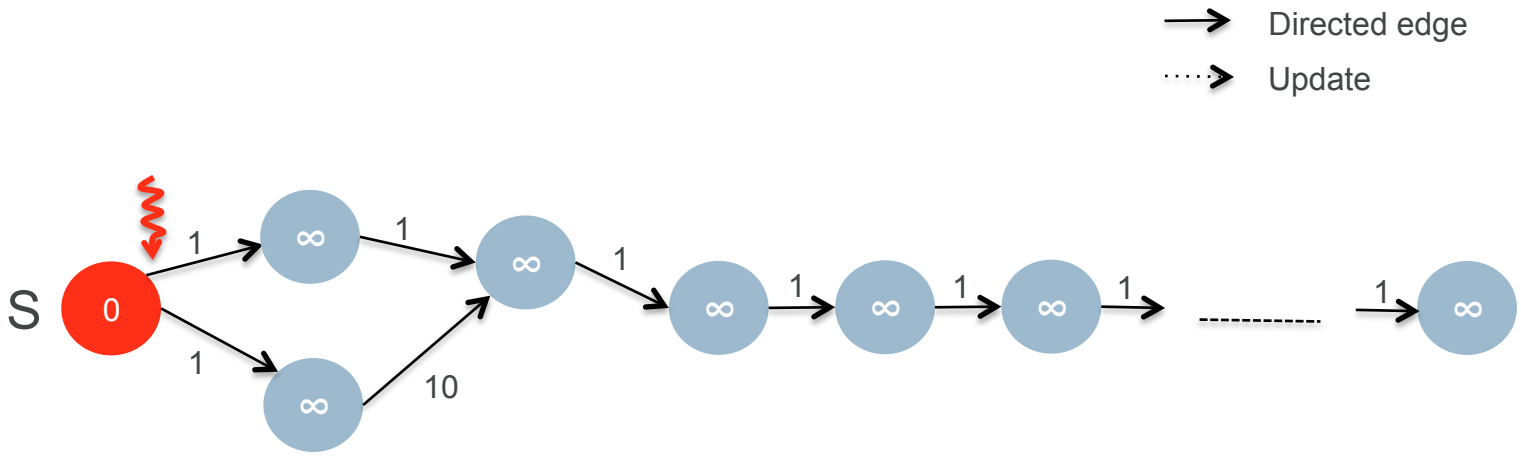
Does not apply to lots of problem classes

Major performance problems

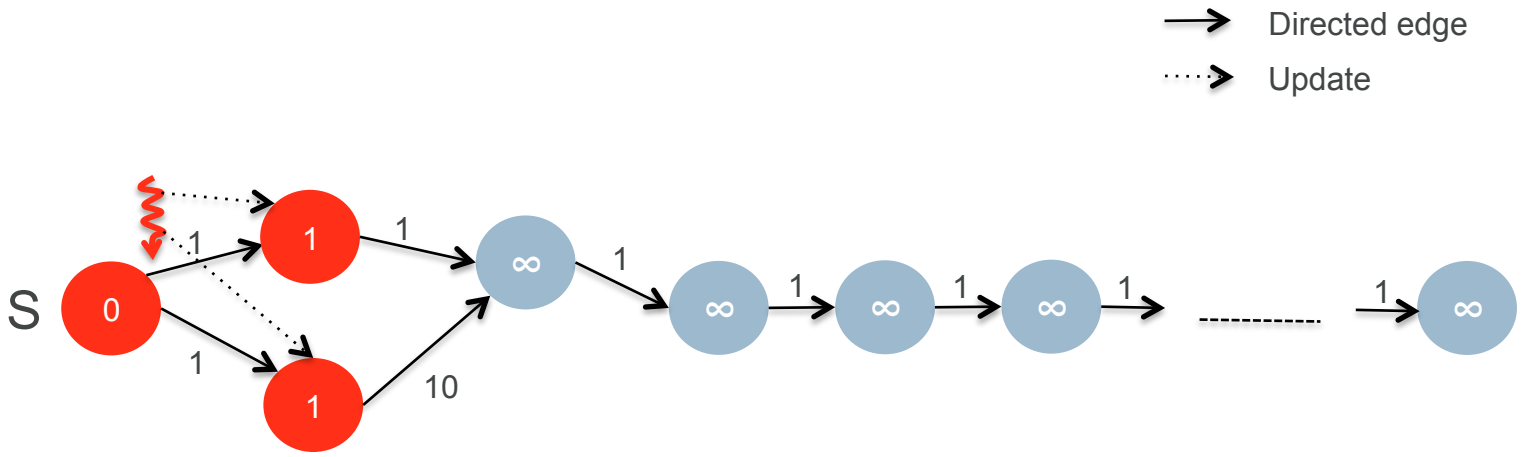
Illustrative Example: Single-Source Shortest Paths (SSSP)



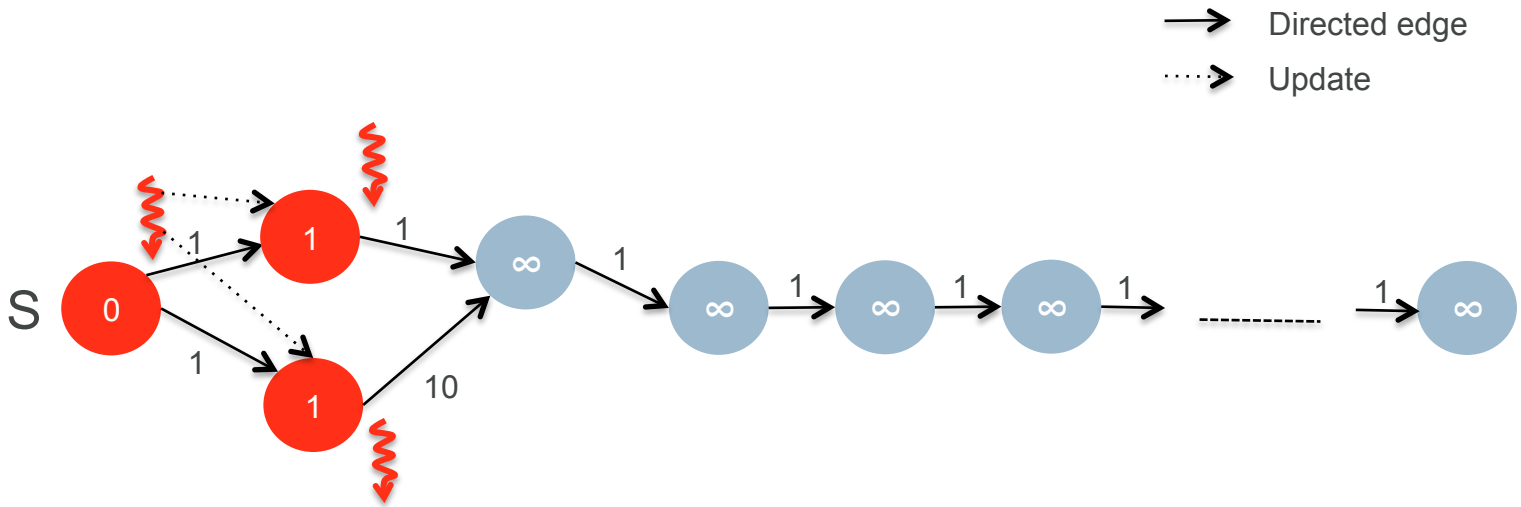
A Data-Driven SSSP algorithm



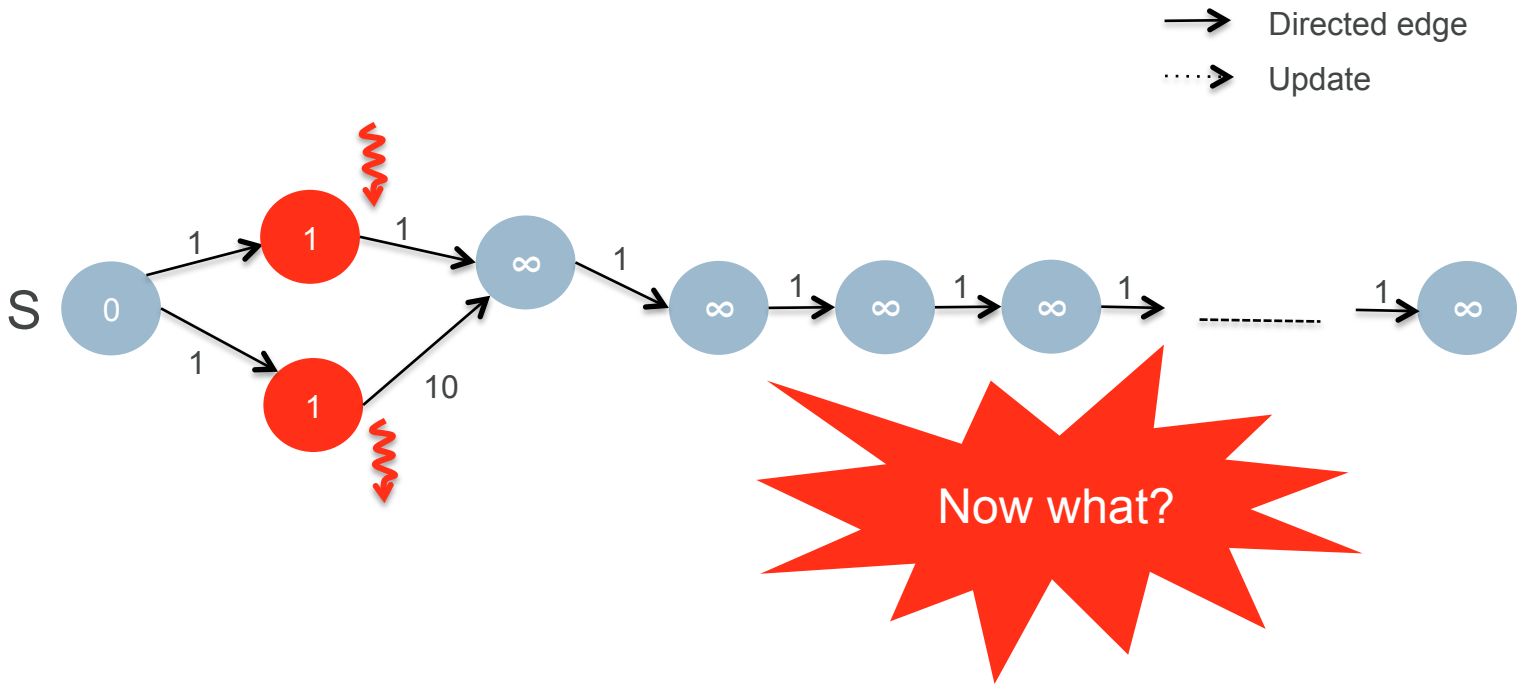
A Data-Driven SSSP algorithm



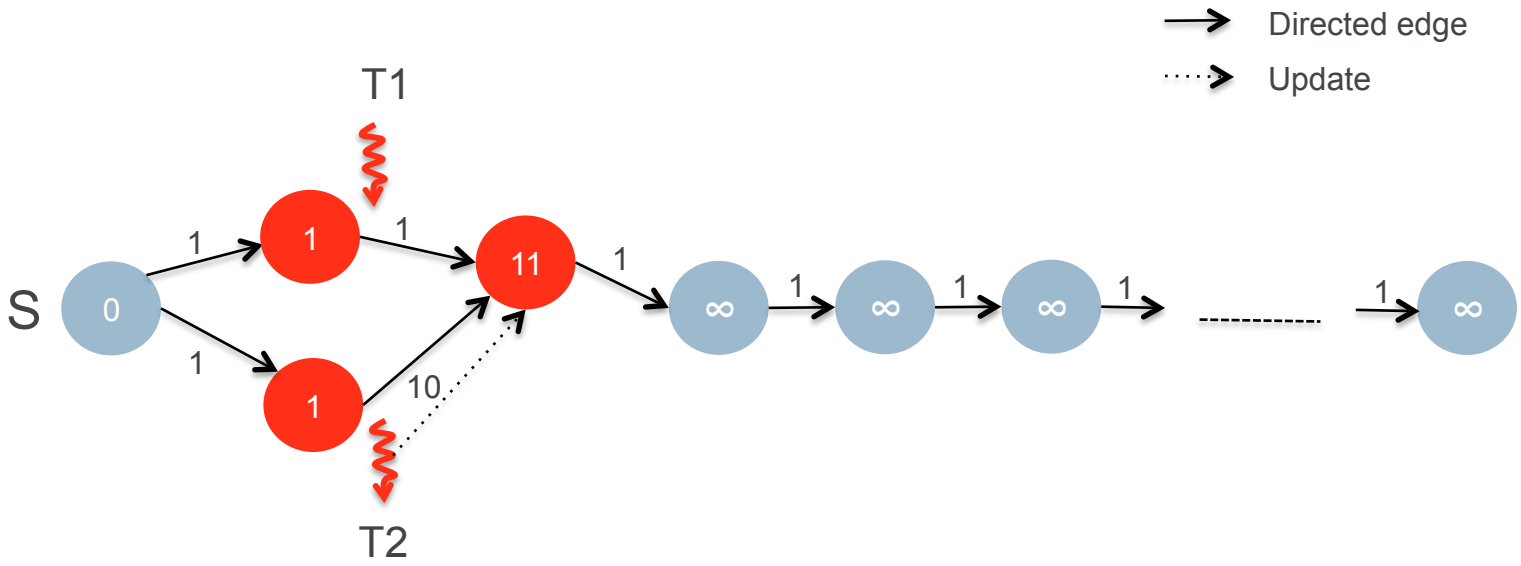
A Data-Driven SSSP algorithm



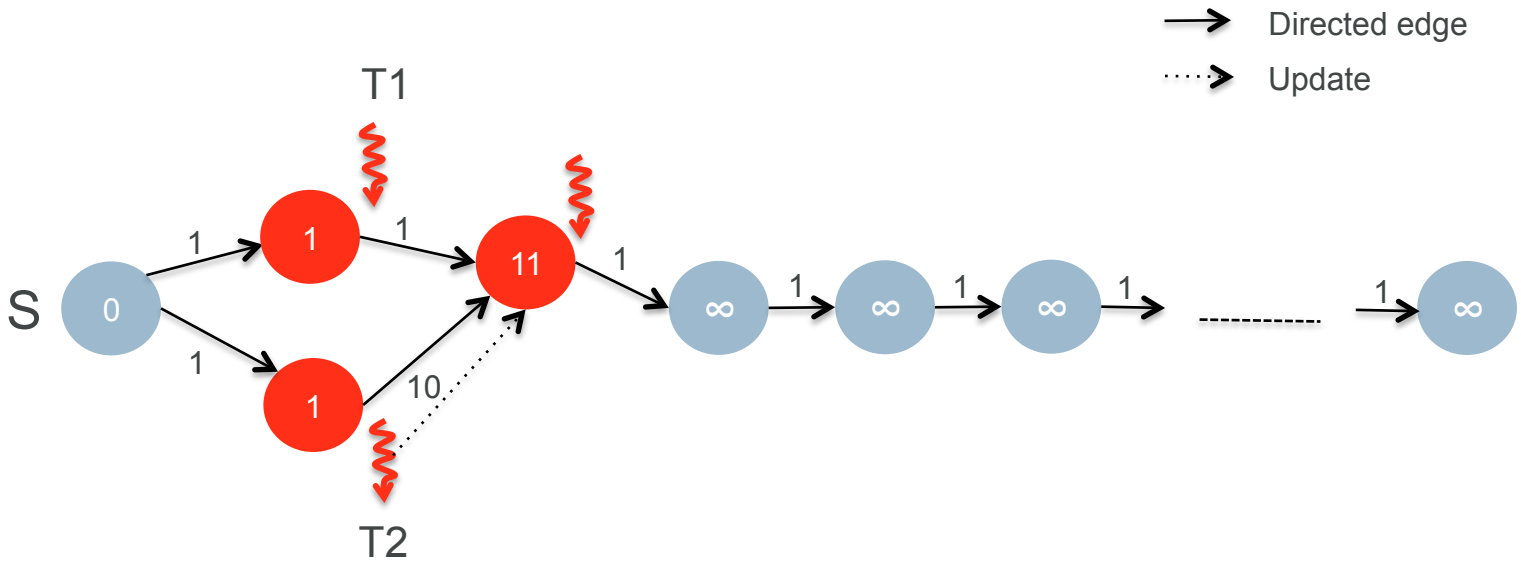
A Data-Driven SSSP algorithm



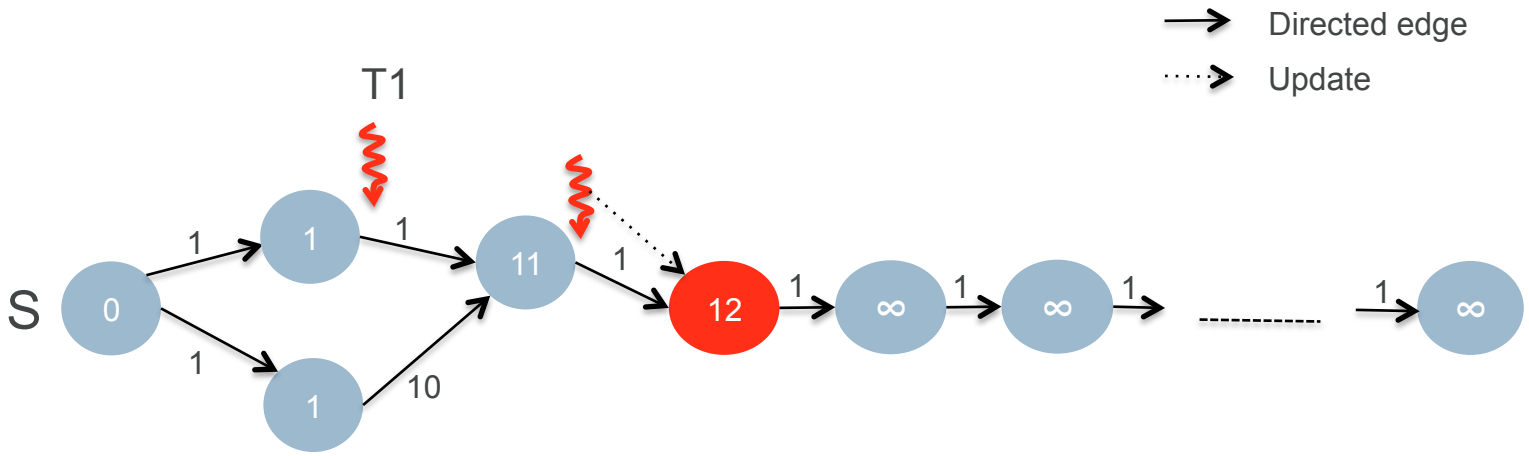
A Data-Driven SSSP algorithm



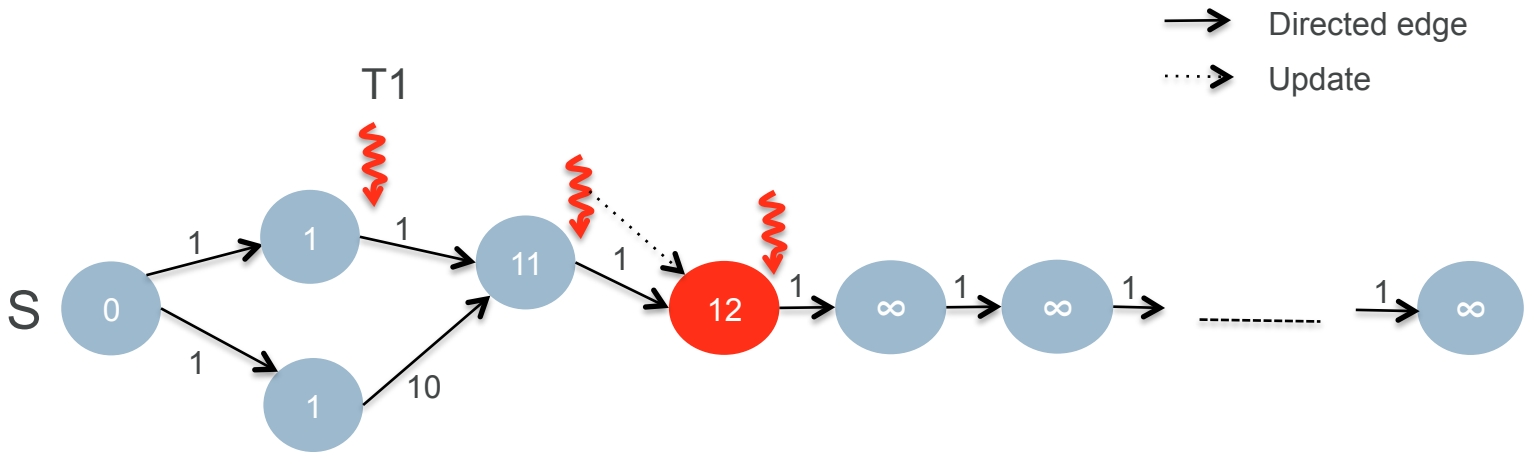
A Data-Driven SSSP algorithm



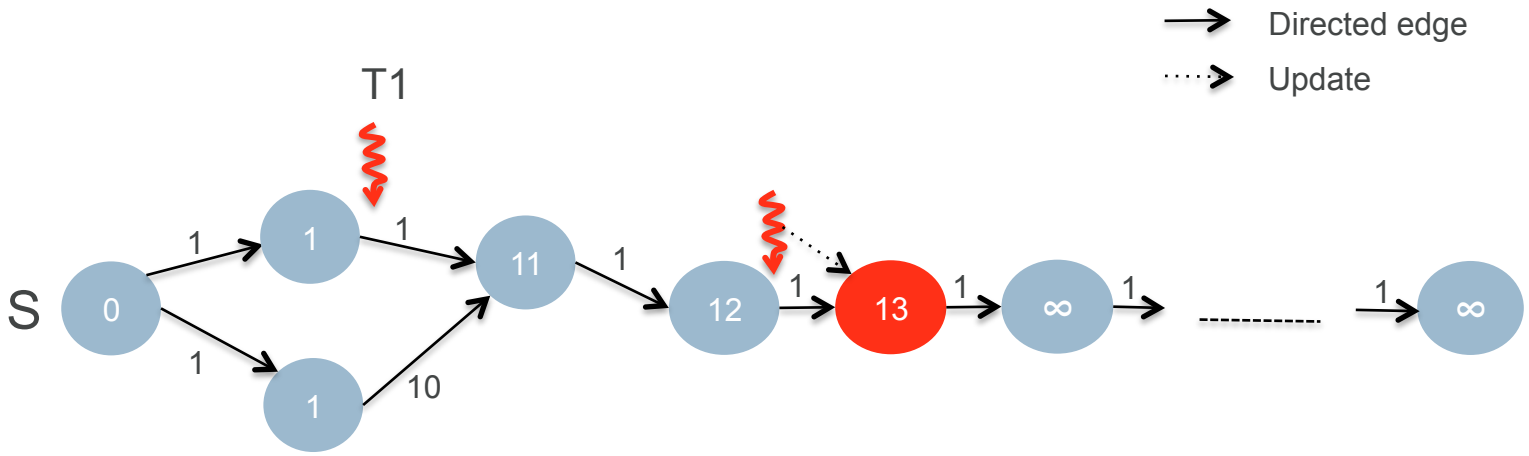
A Data-Driven SSSP algorithm



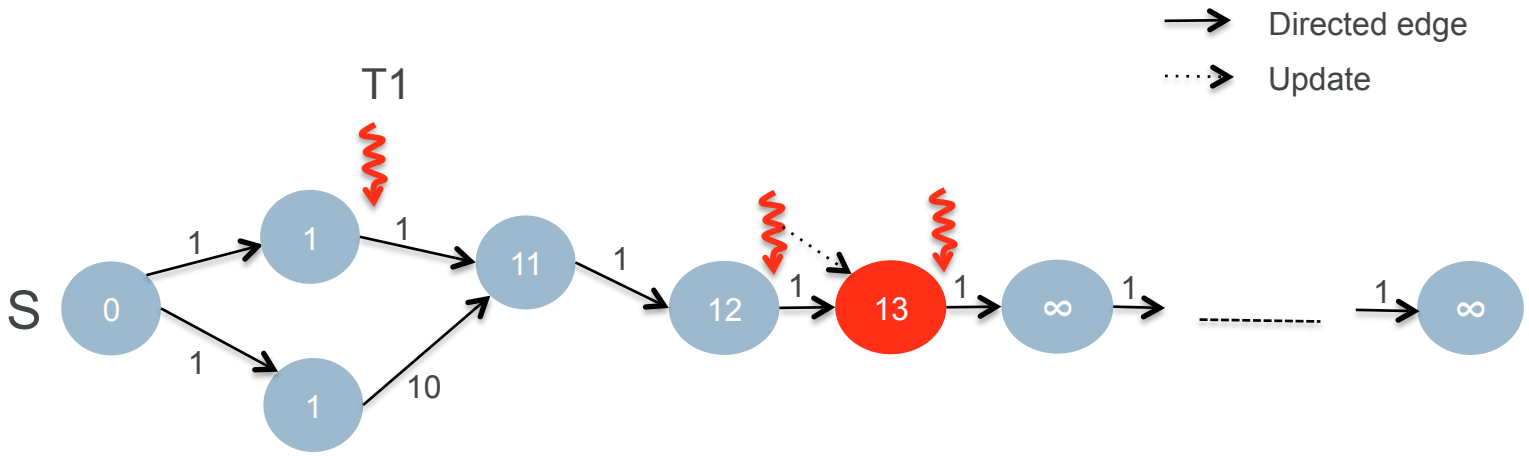
A Data-Driven SSSP algorithm



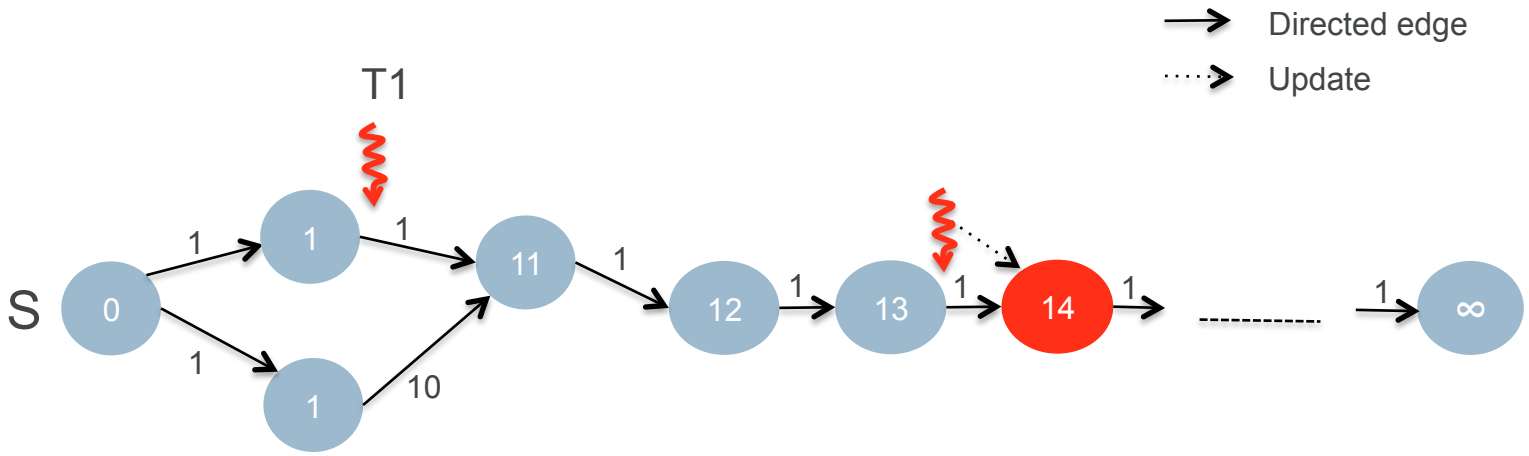
A Data-Driven SSSP algorithm



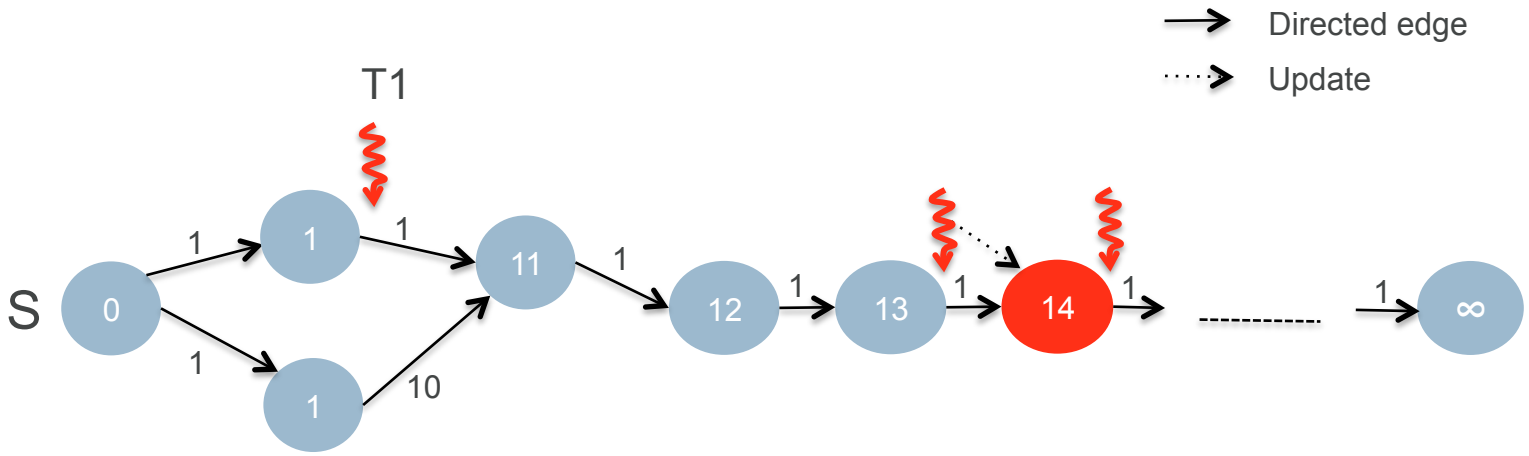
A Data-Driven SSSP algorithm



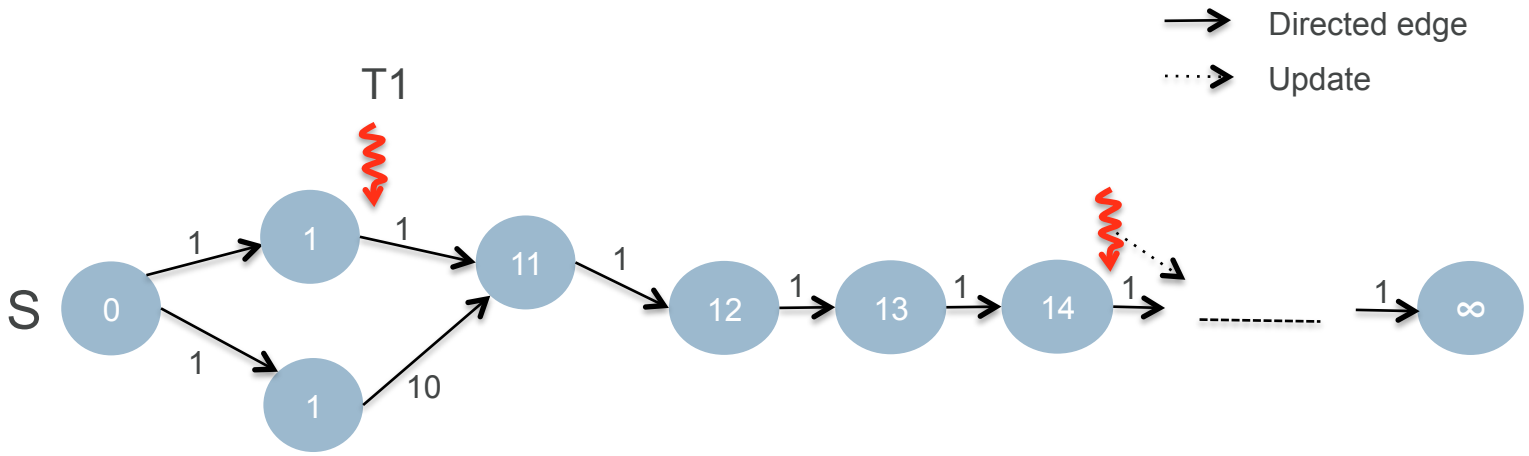
A Data-Driven SSSP algorithm



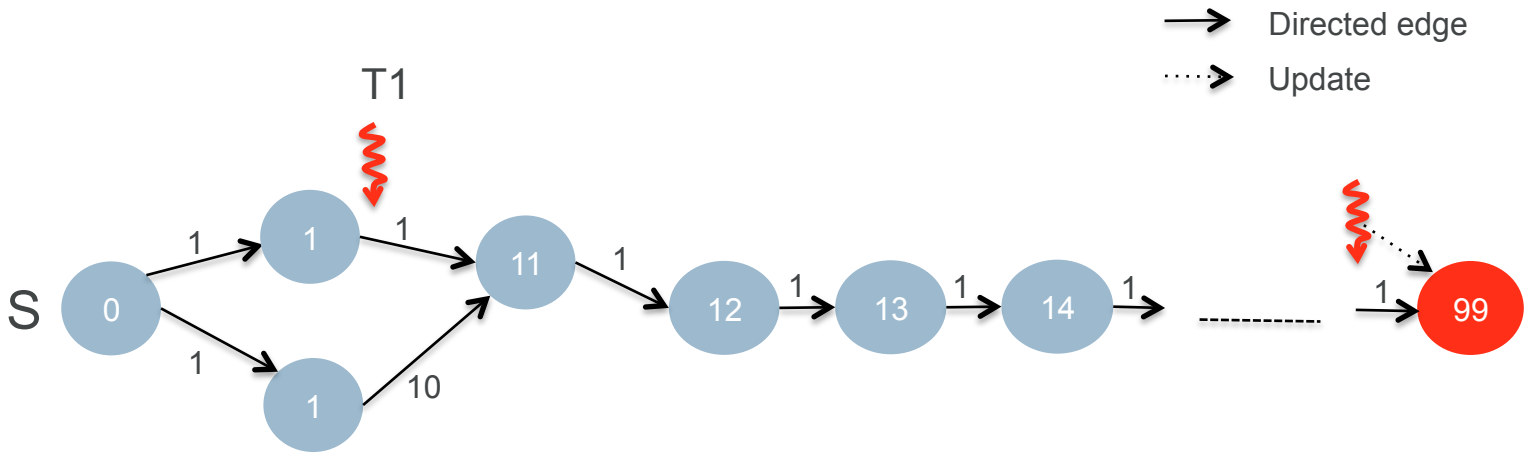
A Data-Driven SSSP algorithm



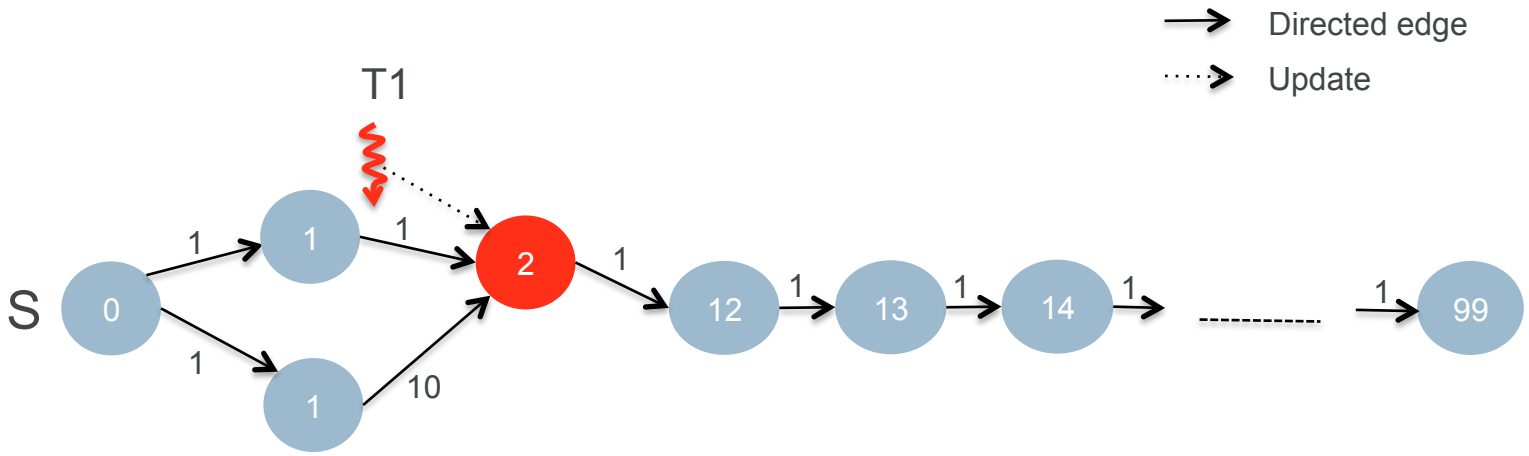
A Data-Driven SSSP algorithm



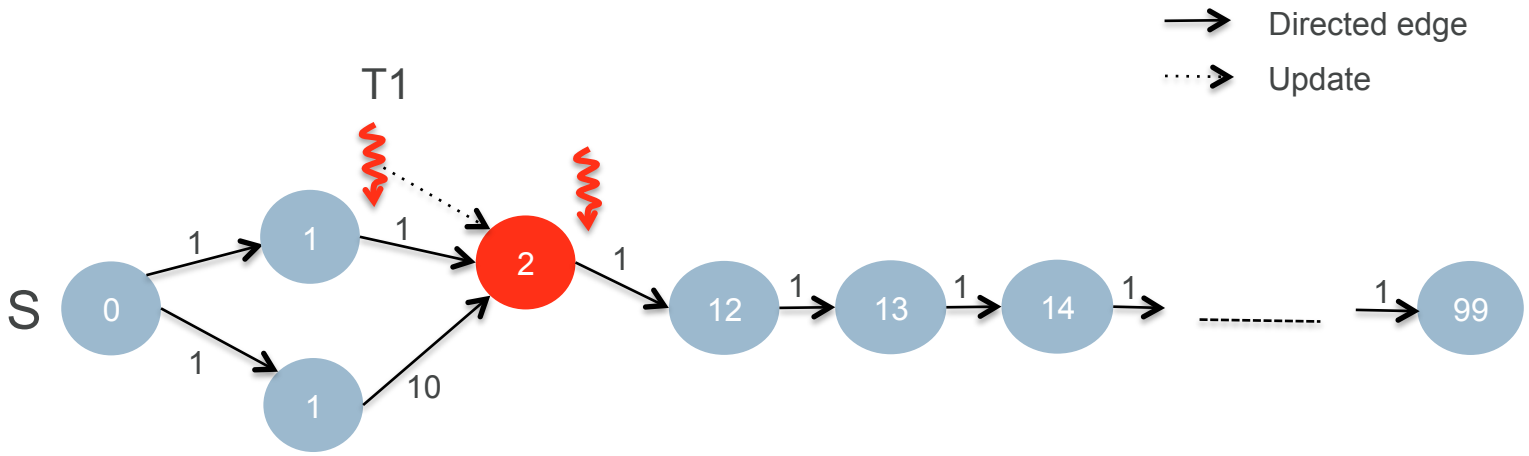
A Data-Driven SSSP algorithm



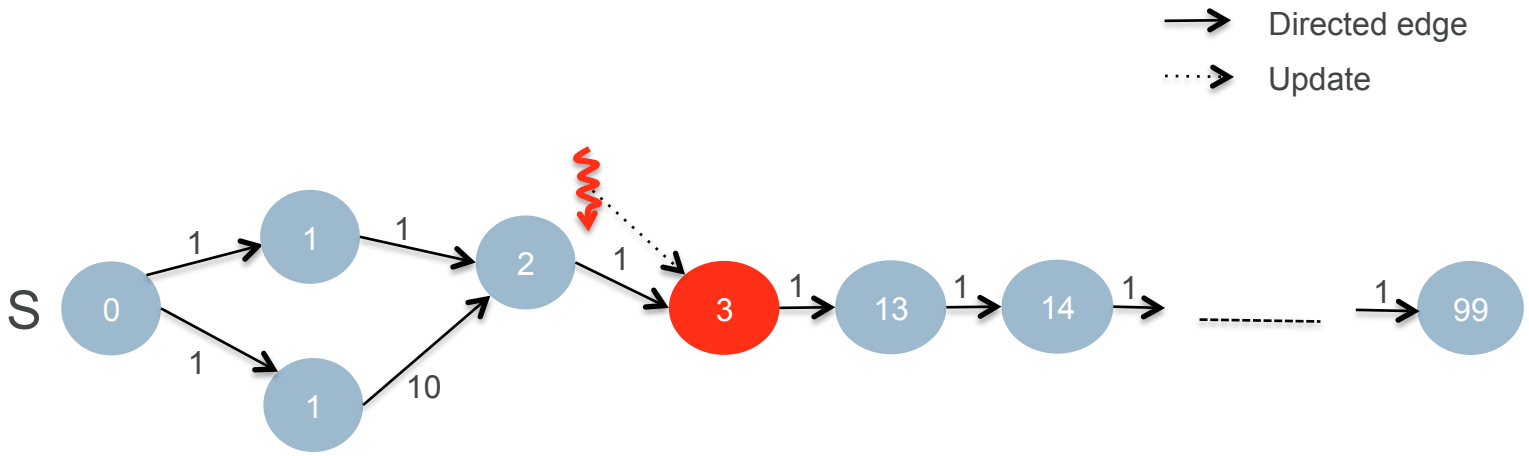
A Data-Driven SSSP algorithm



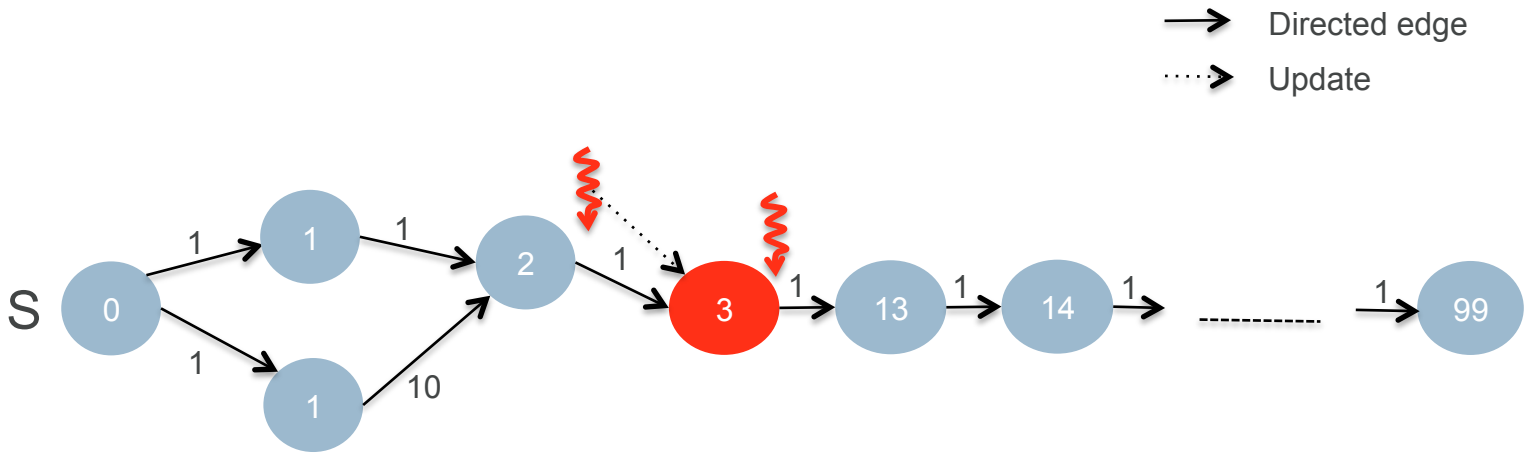
A Data-Driven SSSP algorithm



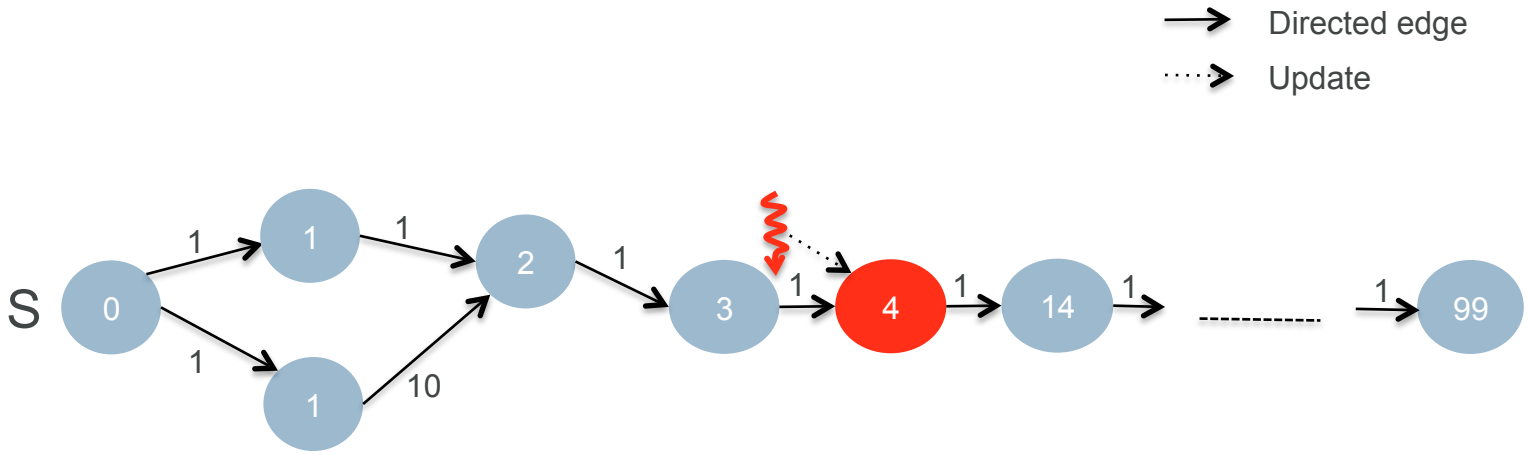
A Data-Driven SSSP algorithm



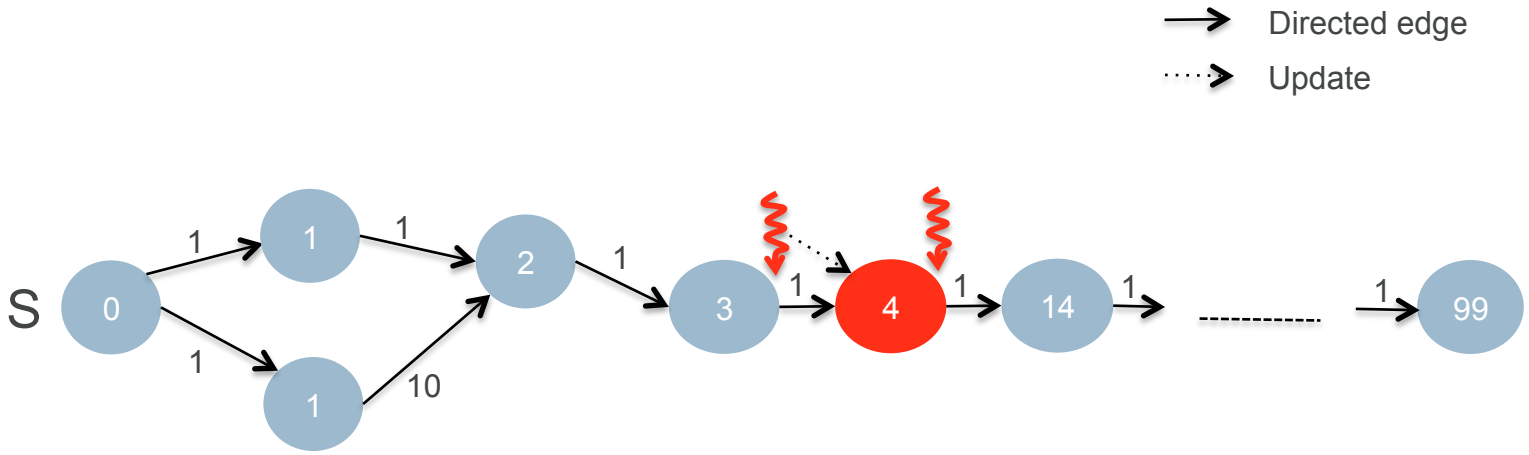
A Data-Driven SSSP algorithm



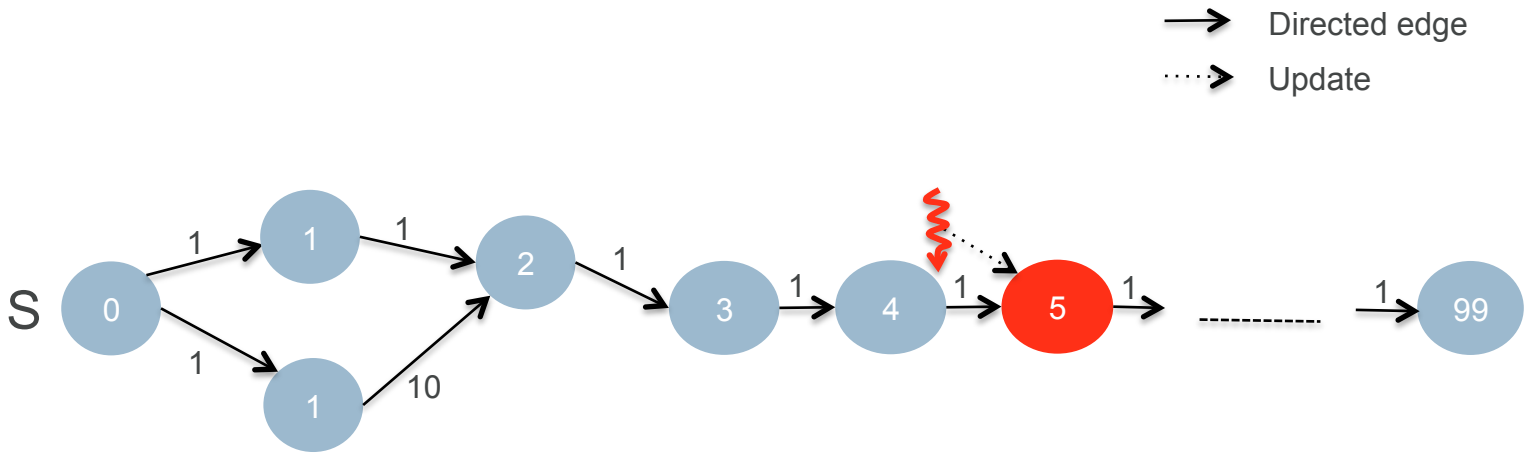
A Data-Driven SSSP algorithm



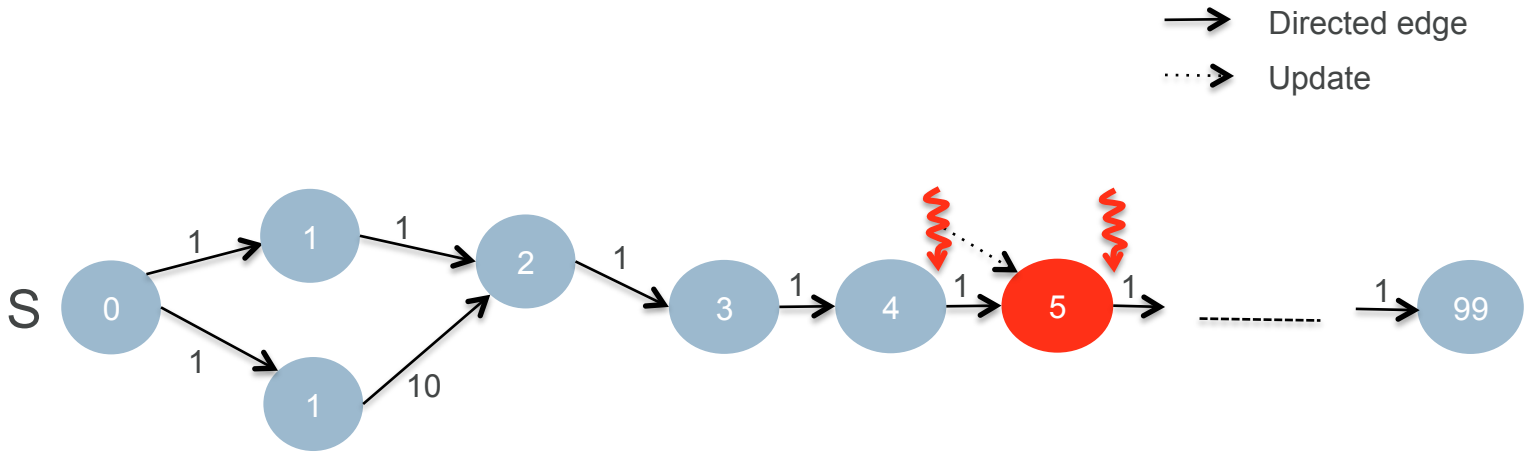
A Data-Driven SSSP algorithm



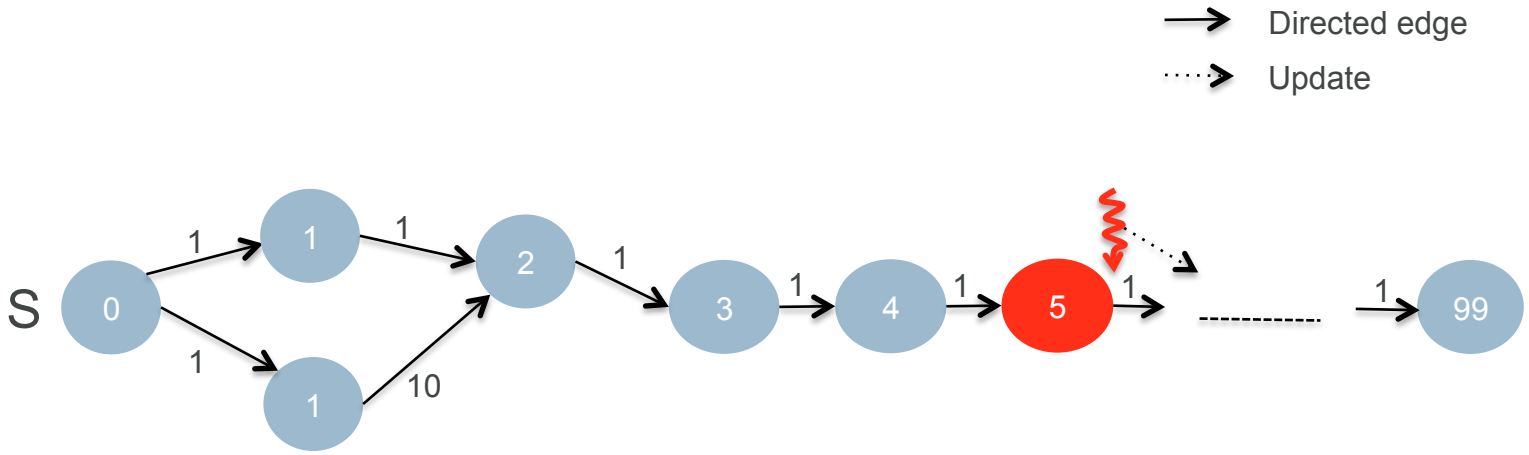
A Data-Driven SSSP algorithm



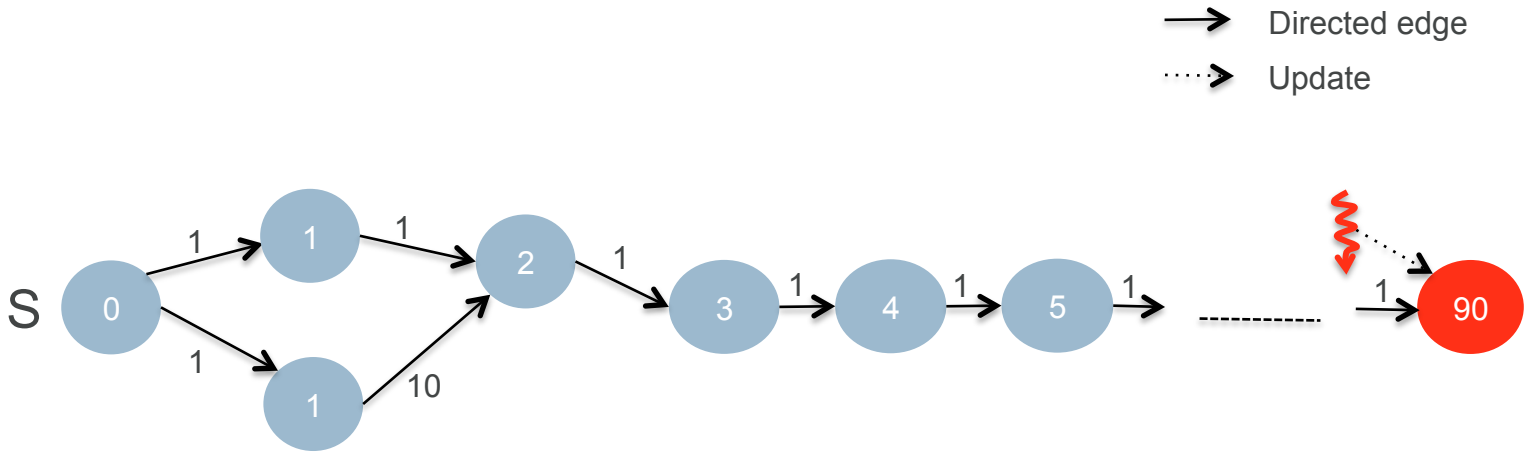
A Data-Driven SSSP algorithm



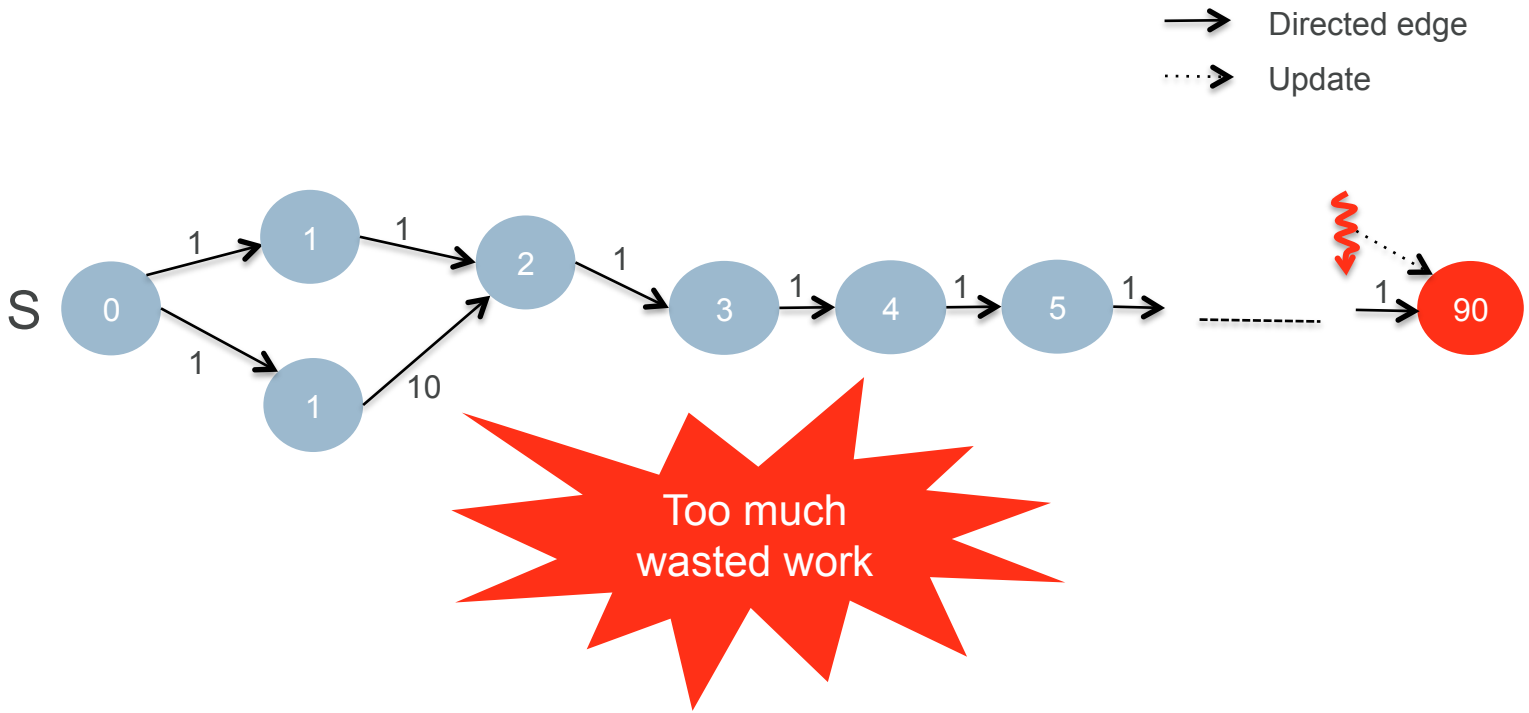
A Data-Driven SSSP algorithm



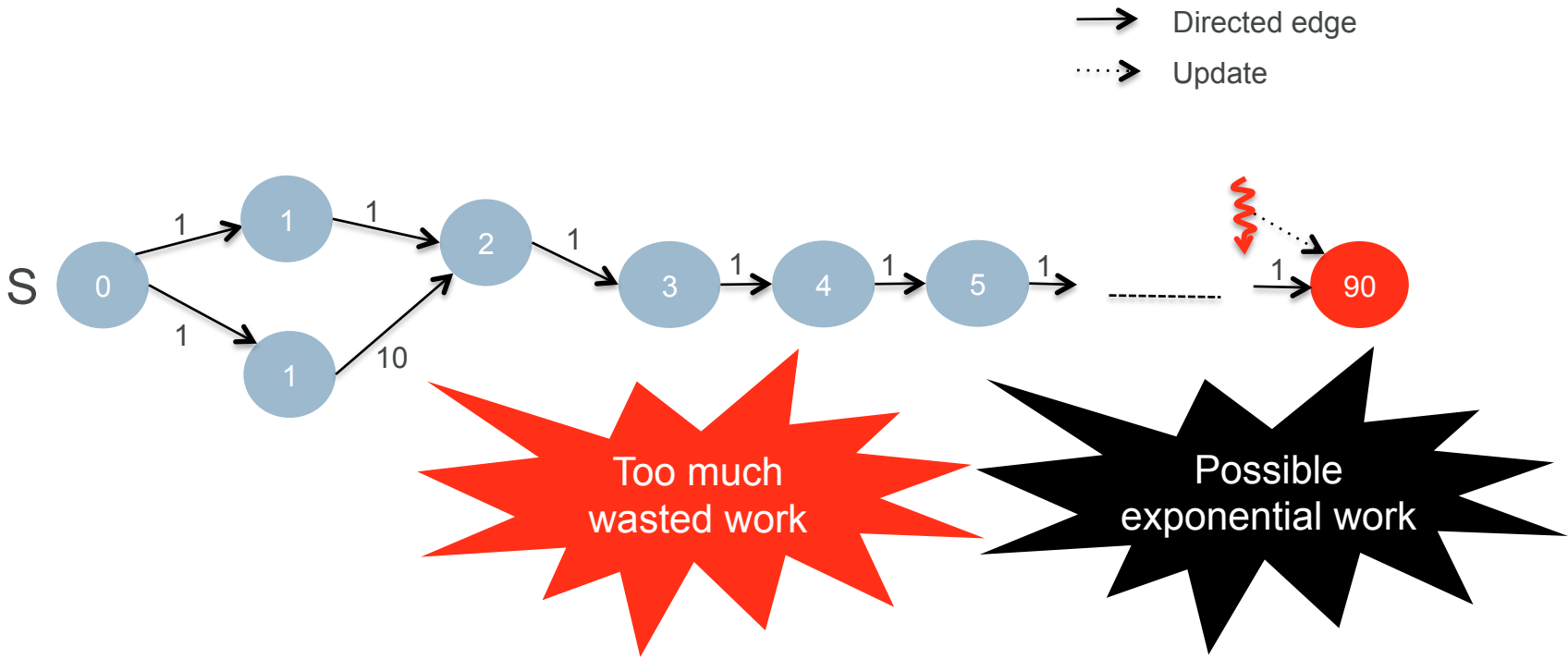
A Data-Driven SSSP algorithm



A Data-Driven SSSP algorithm

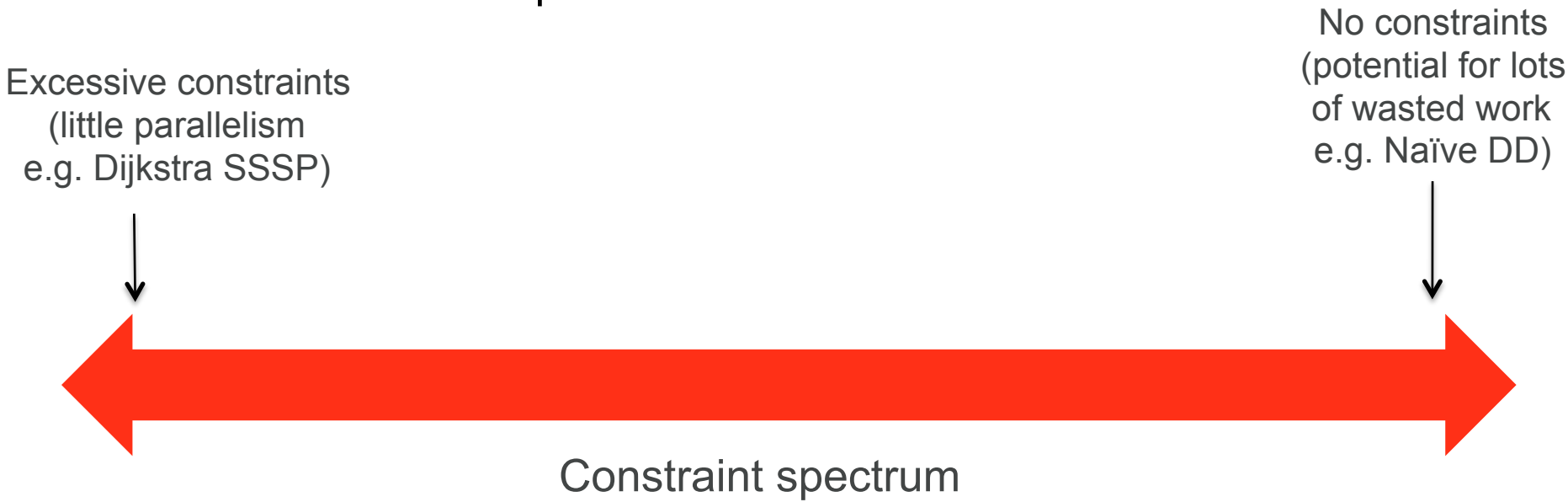


A Data-Driven SSSP algorithm



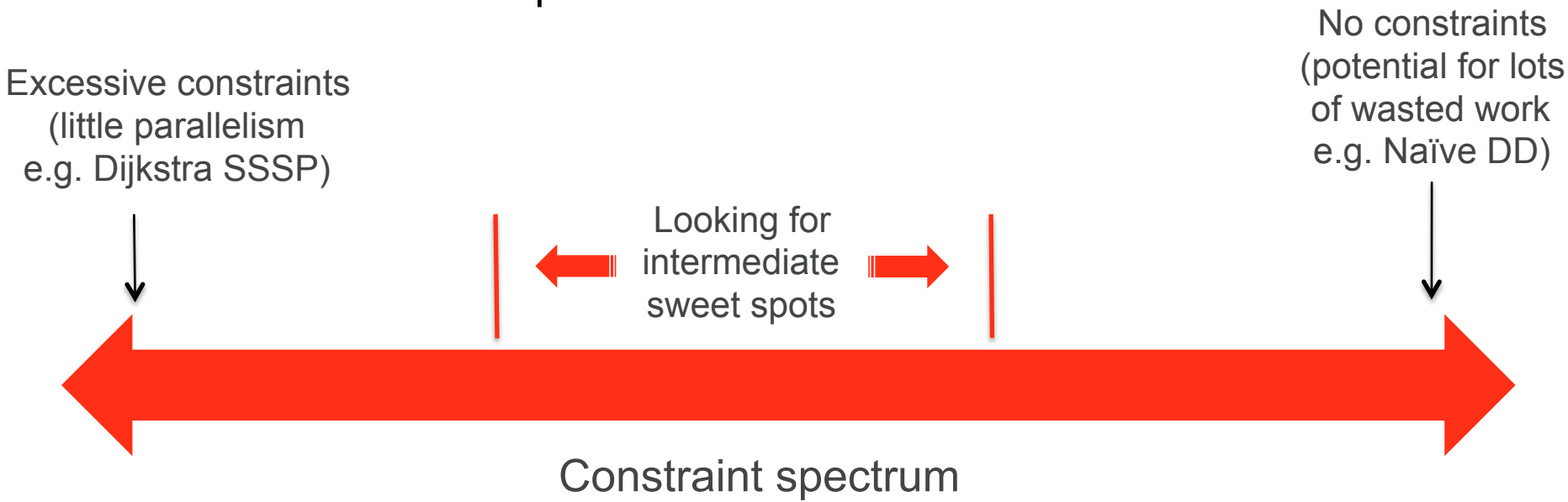
Problem: Execution order of tasks

- Need to *constrain* the execution order
- But “balance” is important



Problem: Execution order of tasks

- Need to *constrain* the execution order
- But “balance” is important



Constrained Data-Driven Parallelism

Abstractions to constrain computation structure

- Task: Schedulable unit of computation
- Triggers: Associate data with triggered function

- ***Phased*** Execution of Tasks:
  - Partitions tasks into a sequence of *phases*
  - Triggers annotated to trigger tasks “right now” or in the “next phase”
  - All the “right now” tasks complete before the “next phase” are scheduled

Constrained Data-Driven Parallelism

Language Constructs

```
1: int * a;
2: *a triggers [deferred] foo();

3: …
4: *a = …; // triggers a task that runs foo()
5: …

6: WaitForTasks(); // wait for all tasks to complete
```

Constrained Data-Driven Parallelism

Language Constructs

```
1: int * a;
2: *a triggers [deferred] foo();

3: …
4: *a = …; // triggers a task that runs foo()
5: …

6: WaitForTasks(); // wait for all tasks to complete
```

Library based implementation at present

Data-Driven SSSP Algorithms

Computation on a Vertex

Naïve Data-Driven

```
RelaxNeighbors(Vertex& v)
  for all n in v.neighbors do
    int * dist = &n.dist;
    *dist triggers RelaxNeighbors(n)

    if (*dist > v.dist + weight(v,n))
      *dist = v.dist + weight(v,n)
```

Data-Driven with Deferred Triggering

```
RelaxNeighbors(Vertex& v)
  for all n in v.neighbors do
    int * dist = &n.dist;
    *dist triggers deferred RelaxNeighbors(n)

    if (*dist > v.dist + weight(v,n))
      *dist = v.dist + weight(v,n)
```

Data-Driven SSSP Algorithms

Computation on a Vertex

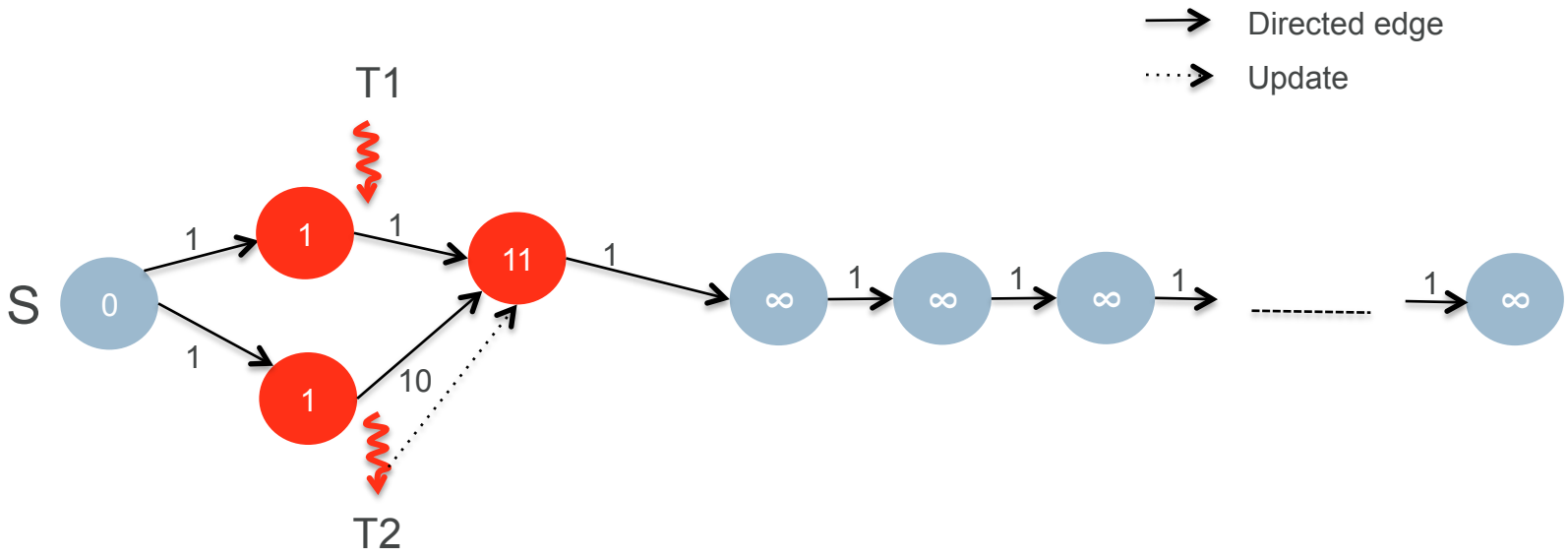
Naïve Data-Driven

```
RelaxNeighbors(Vertex& v)
  for all n in v.neighbors do
    int * dist = &n.dist;
    *dist triggers RelaxNeighbors(n)

    if (*dist > v.dist + weight(v,n))
      *dist = v.dist + weight(v,n)
```

Data-Driven with Deferred Triggering

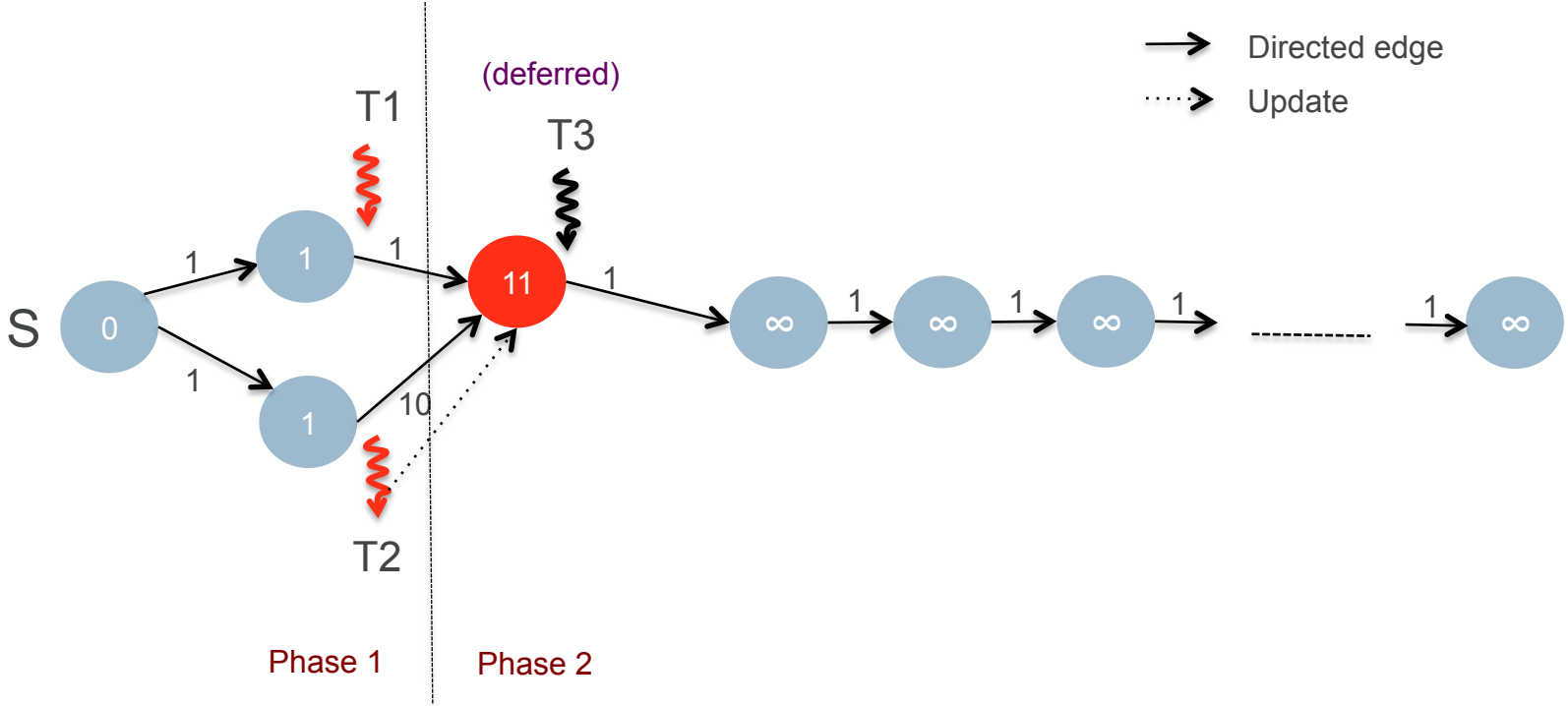
```
RelaxNeighbors(Vertex& v)
  for all n in v.neighbors do
    int * dist = &n.dist;
    *dist triggers deferred RelaxNeighbors(n)

    if (*dist > v.dist + weight(v,n))
      *dist = v.dist + weight(v,n)
```

A Data-Driven SSSP algorithm (with deferred triggering)

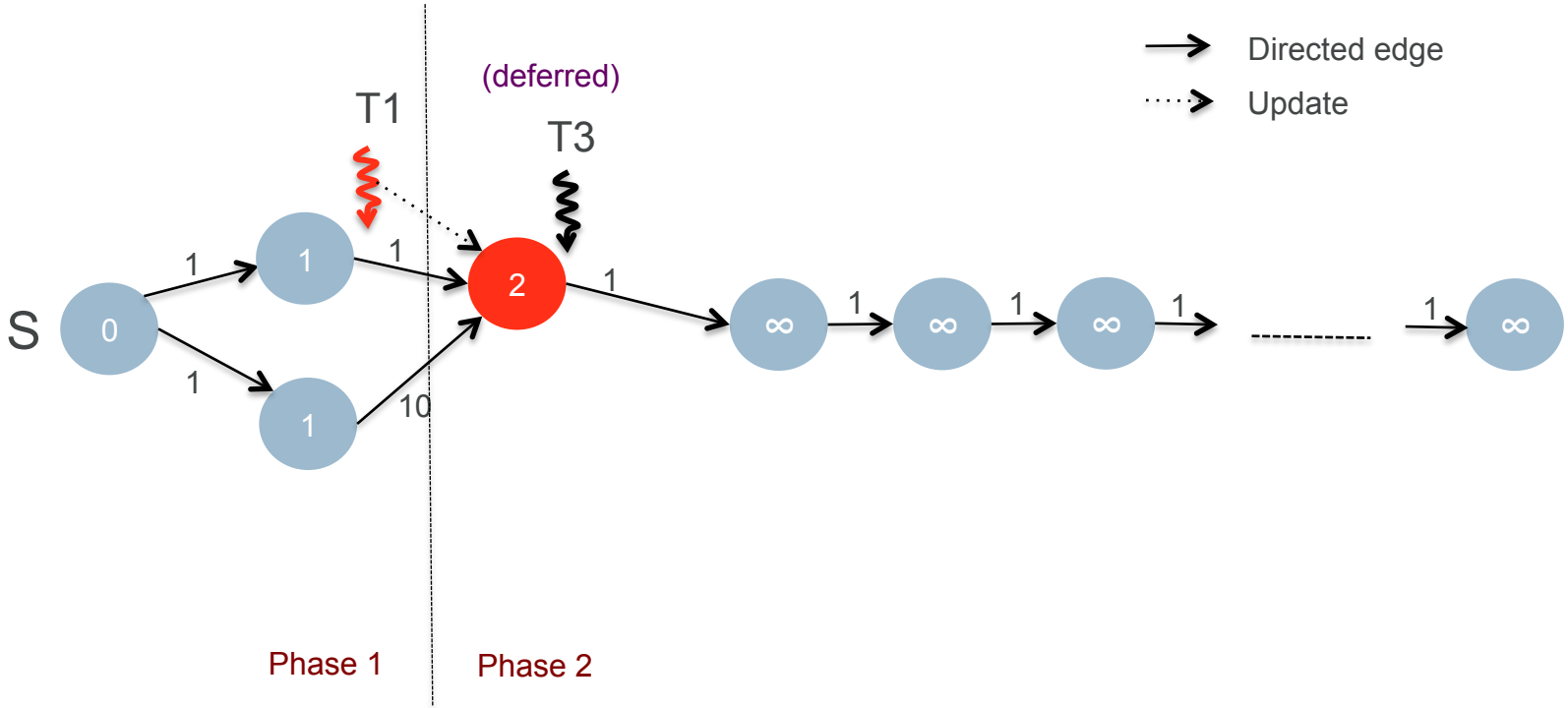


Phase 1

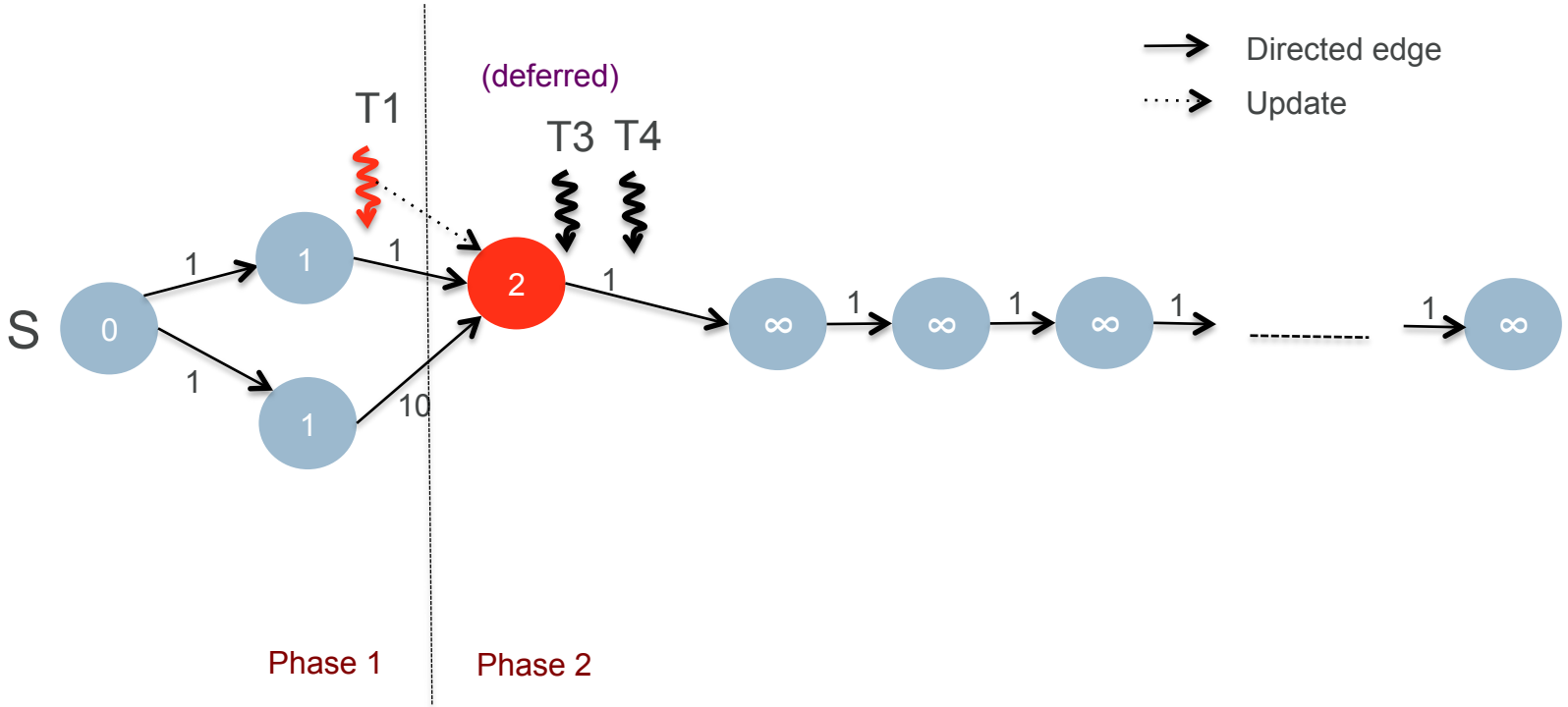
A Data-Driven SSSP algorithm (with deferred triggering)



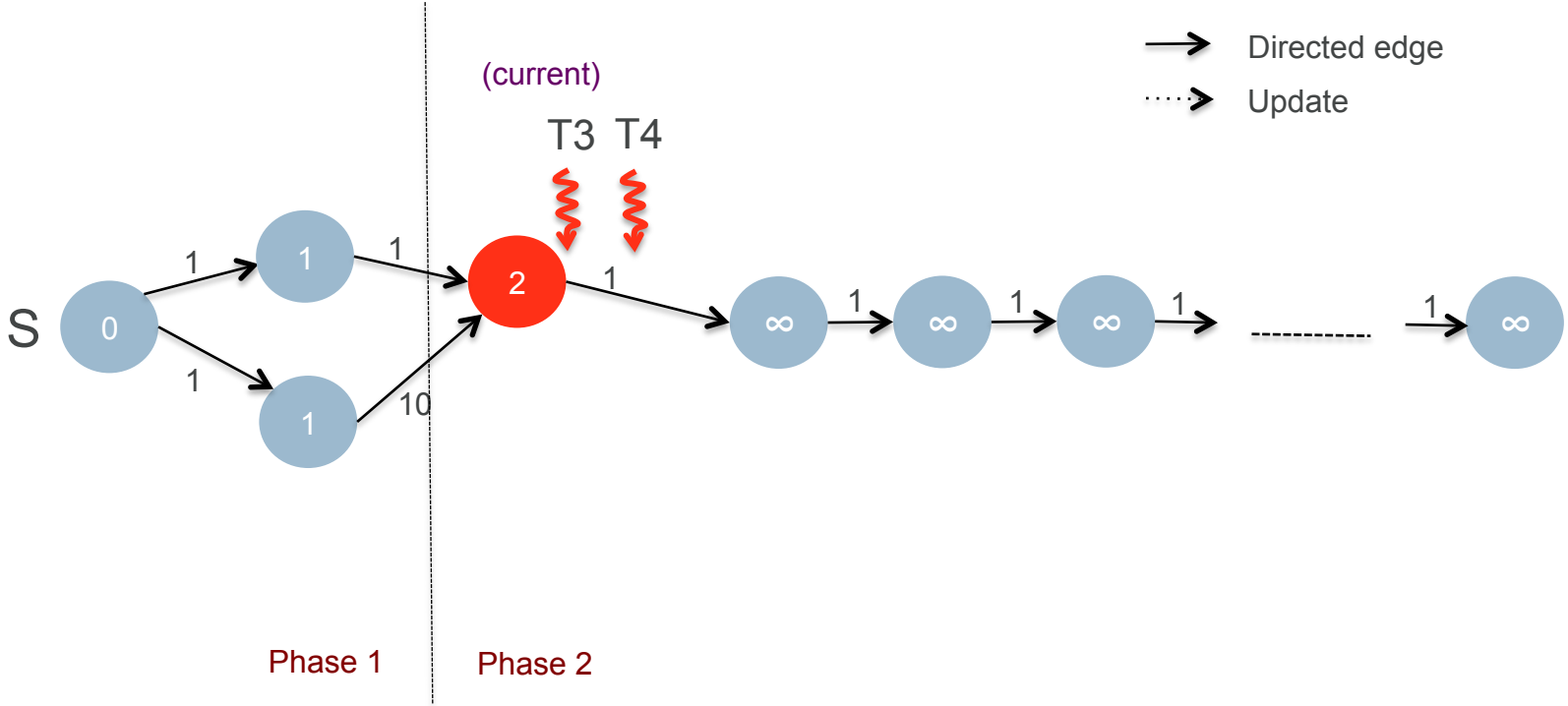
A Data-Driven SSSP algorithm (with deferred triggering)



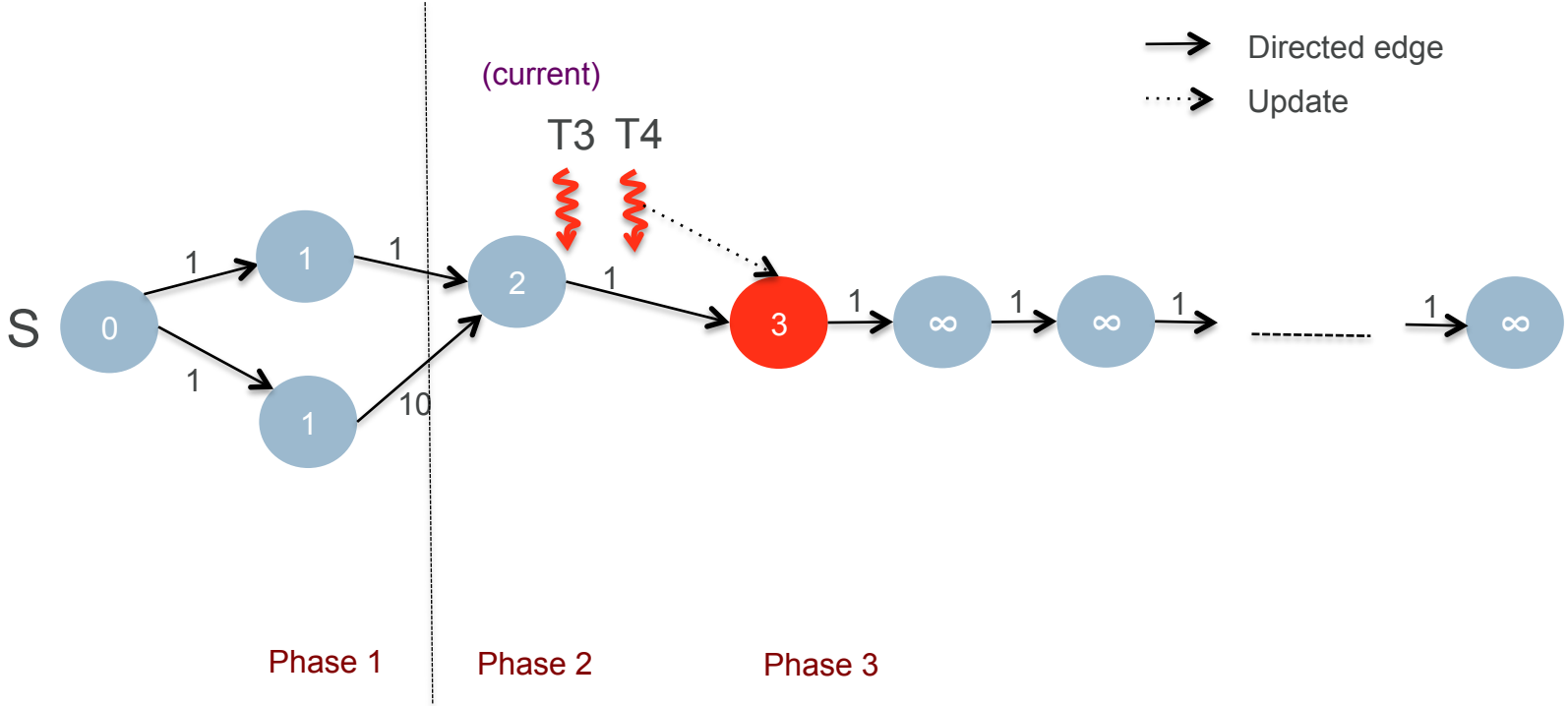
A Data-Driven SSSP algorithm (with deferred triggering)



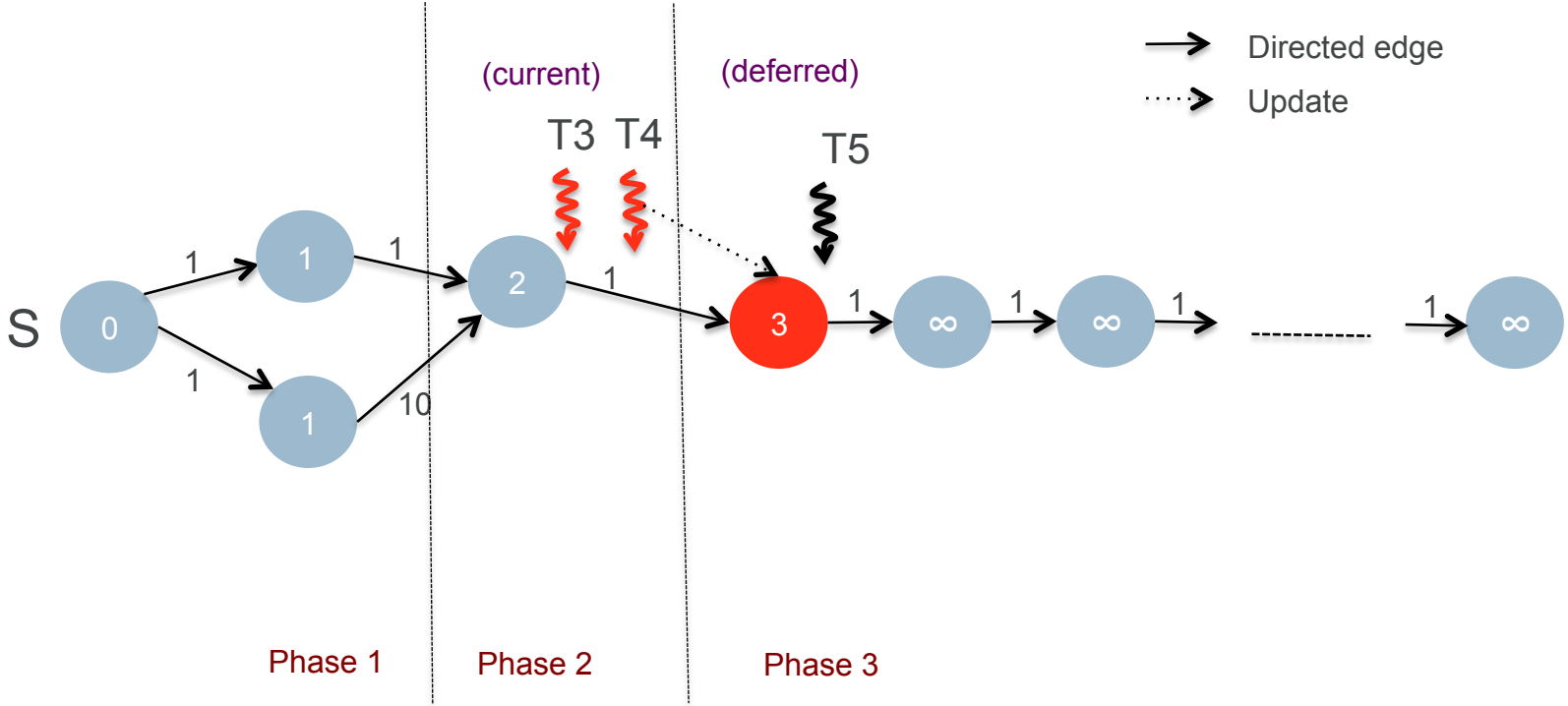
A Data-Driven SSSP algorithm (with deferred triggering)



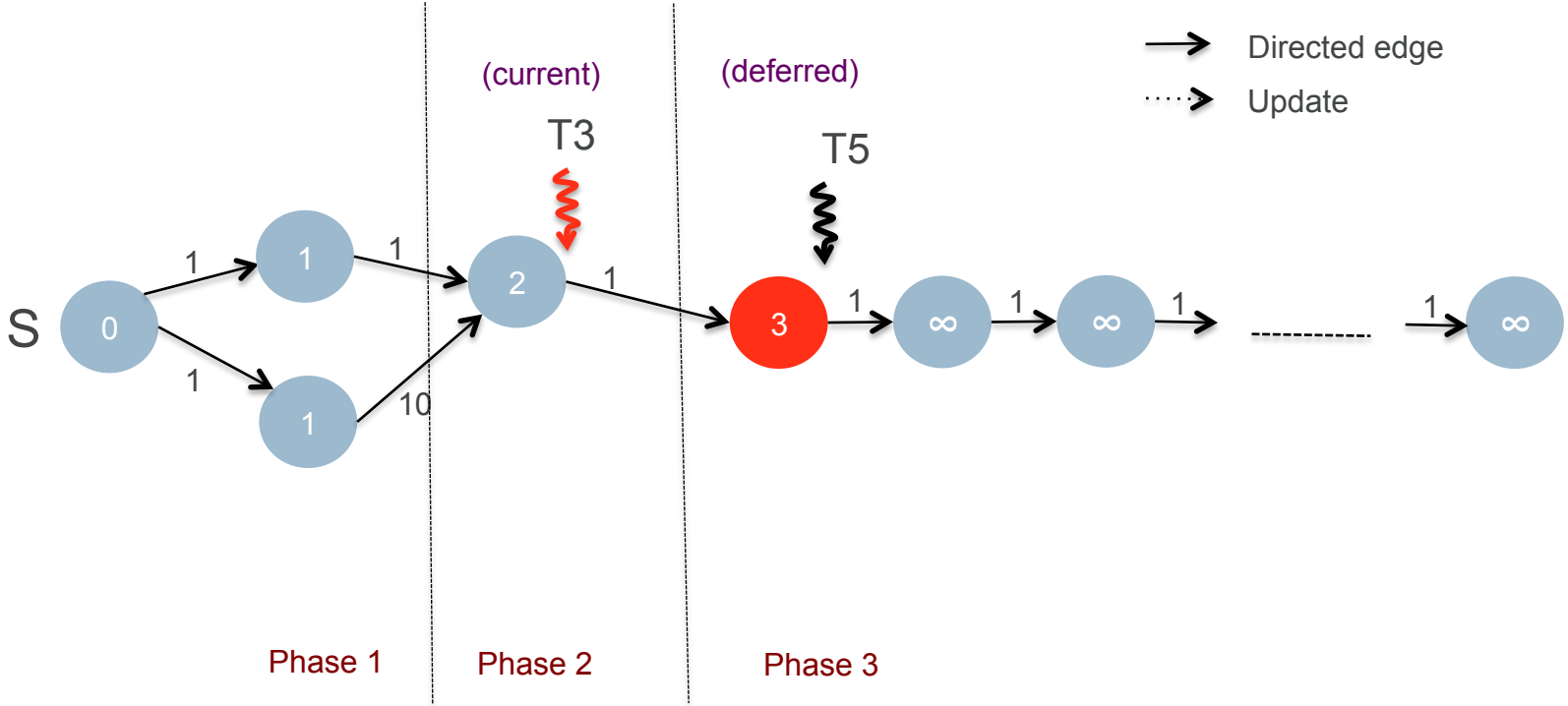
A Data-Driven SSSP algorithm (with deferred triggering)



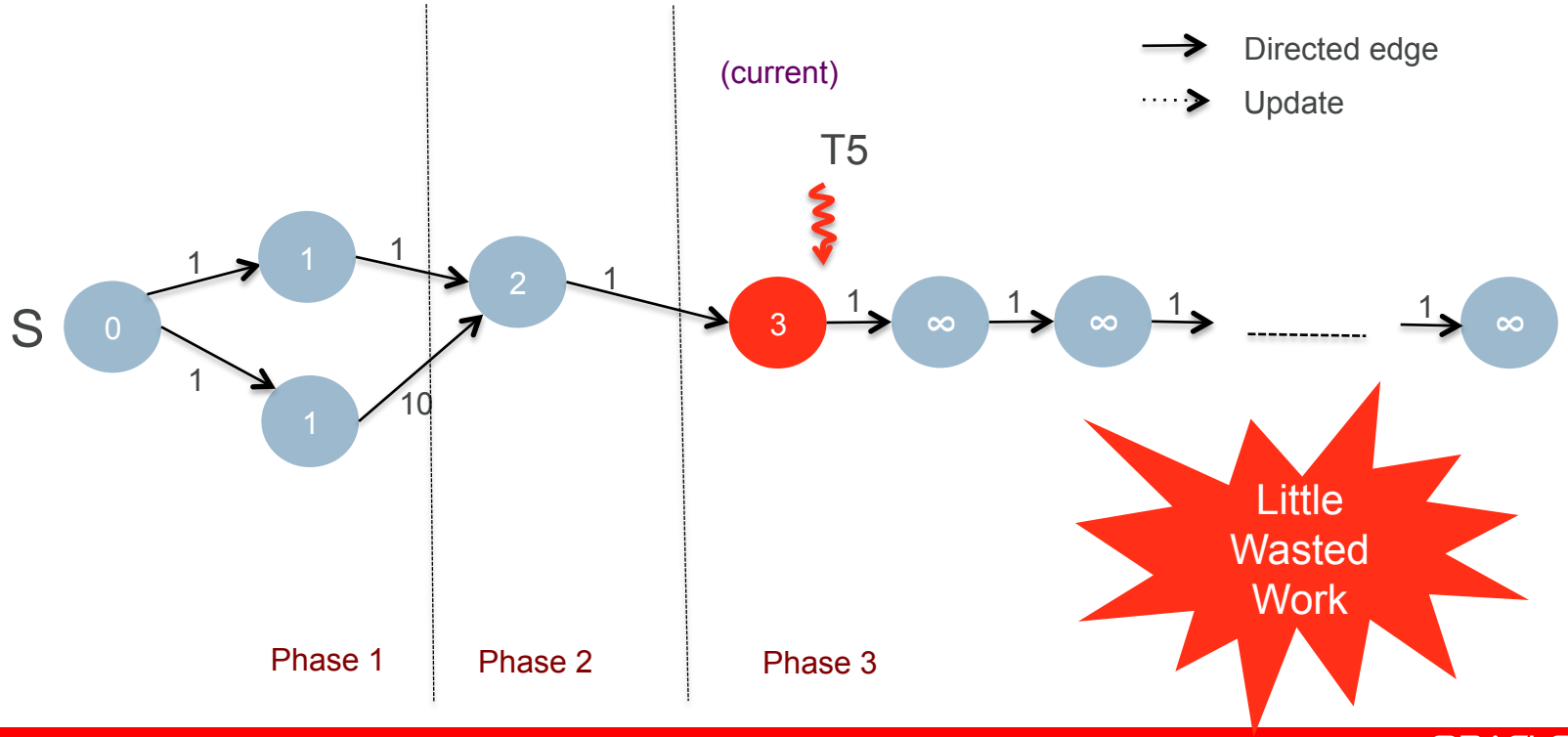
A Data-Driven SSSP algorithm (with deferred triggering)



A Data-Driven SSSP algorithm (with deferred triggering)



A Data-Driven SSSP algorithm (with deferred triggering)



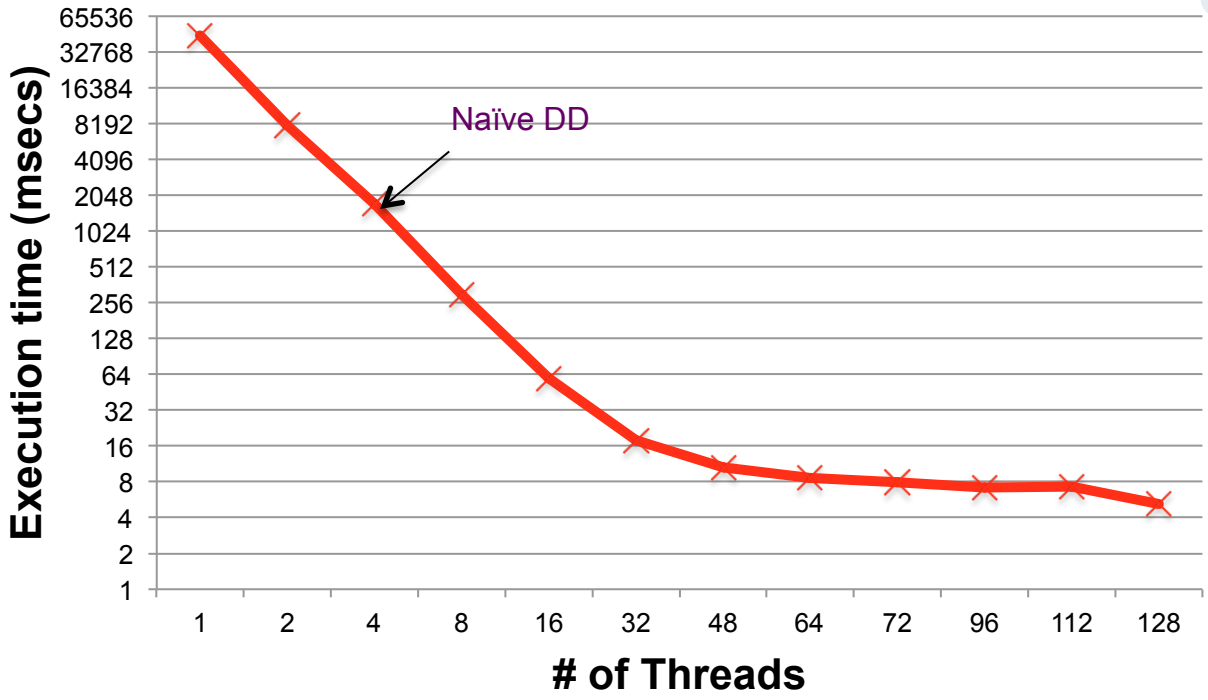
Implementation

- C++ Library Implementation
- Work-stealing based Task Scheduler

SSSP Scalability Results

Architecture:
128-thread 2-socket SPARC T2+ system

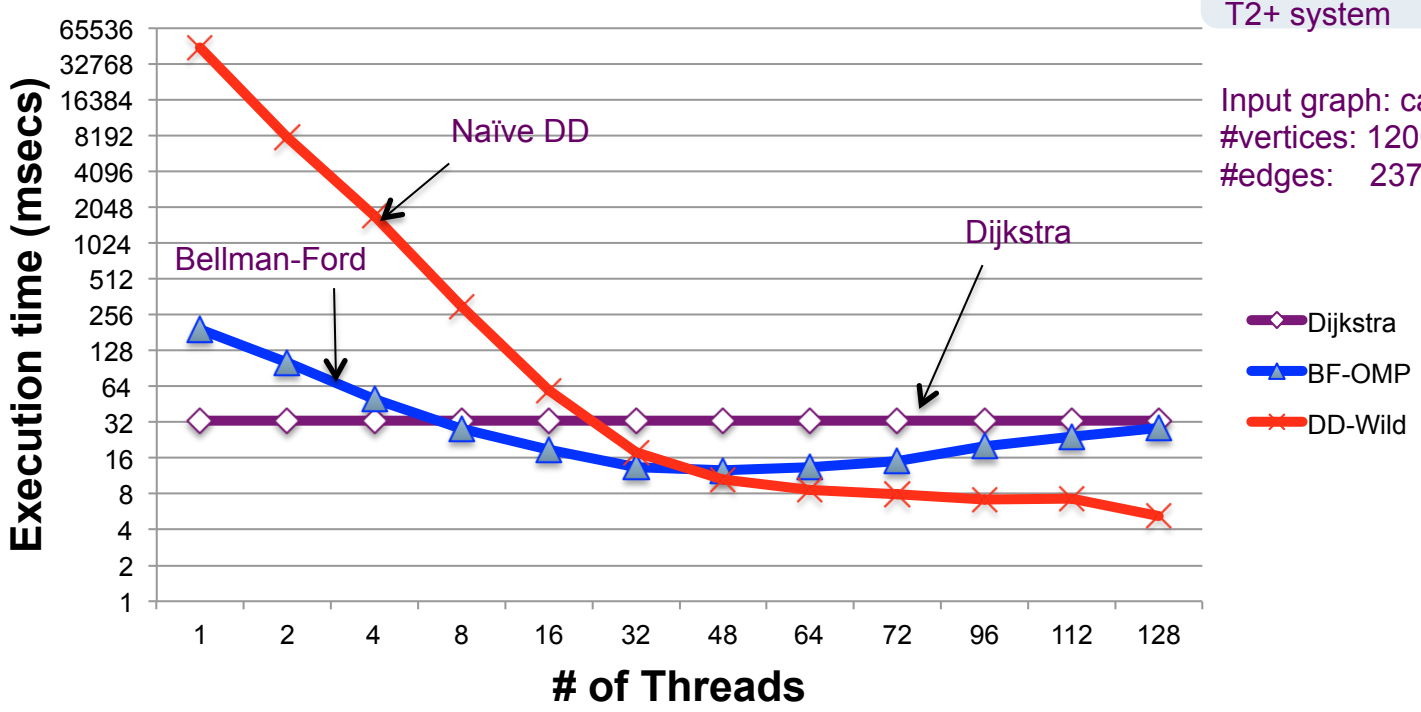
Input graph: ca-HepPh
#vertices: 12008
#edges: 237042



SSSP Scalability Results

Architecture:
128-thread 2-socket SPARC T2+ system

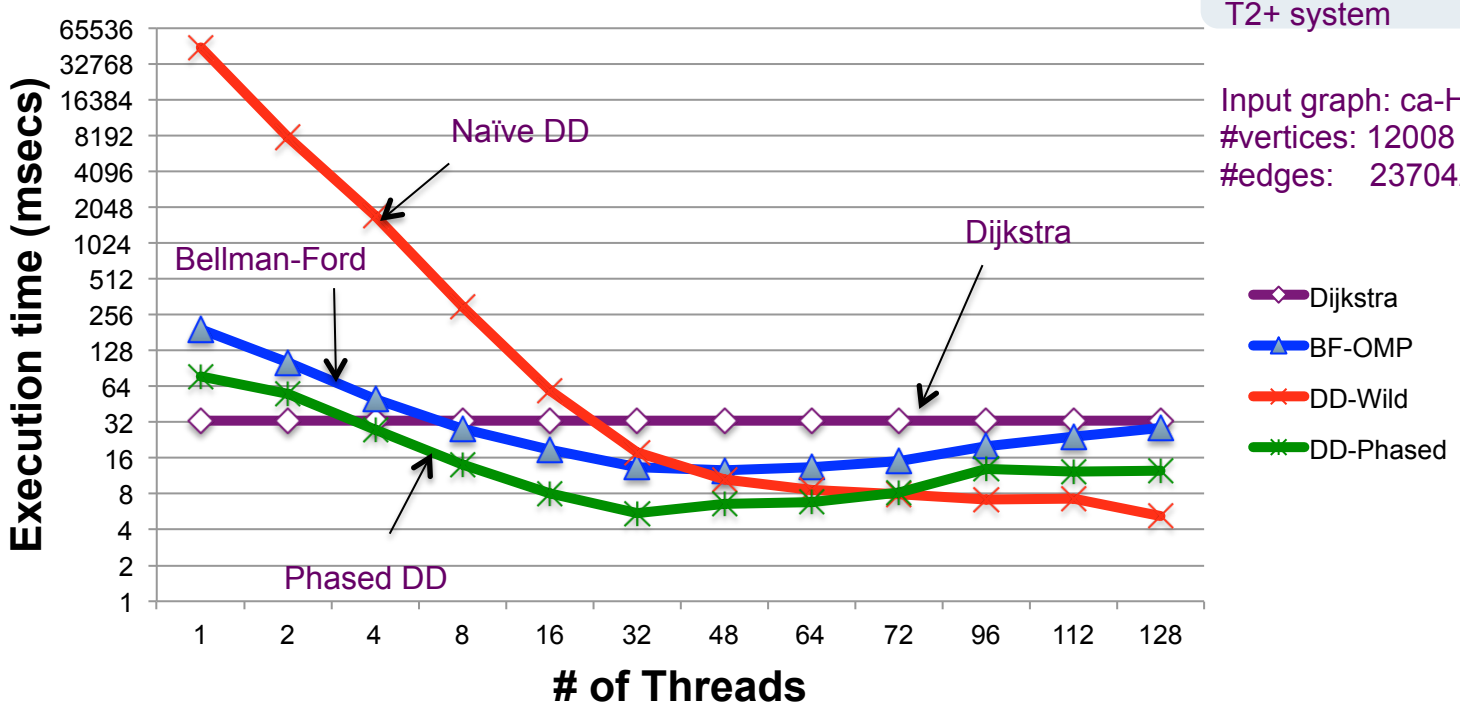
Input graph: ca-HepPh
#vertices: 12008
#edges: 237042



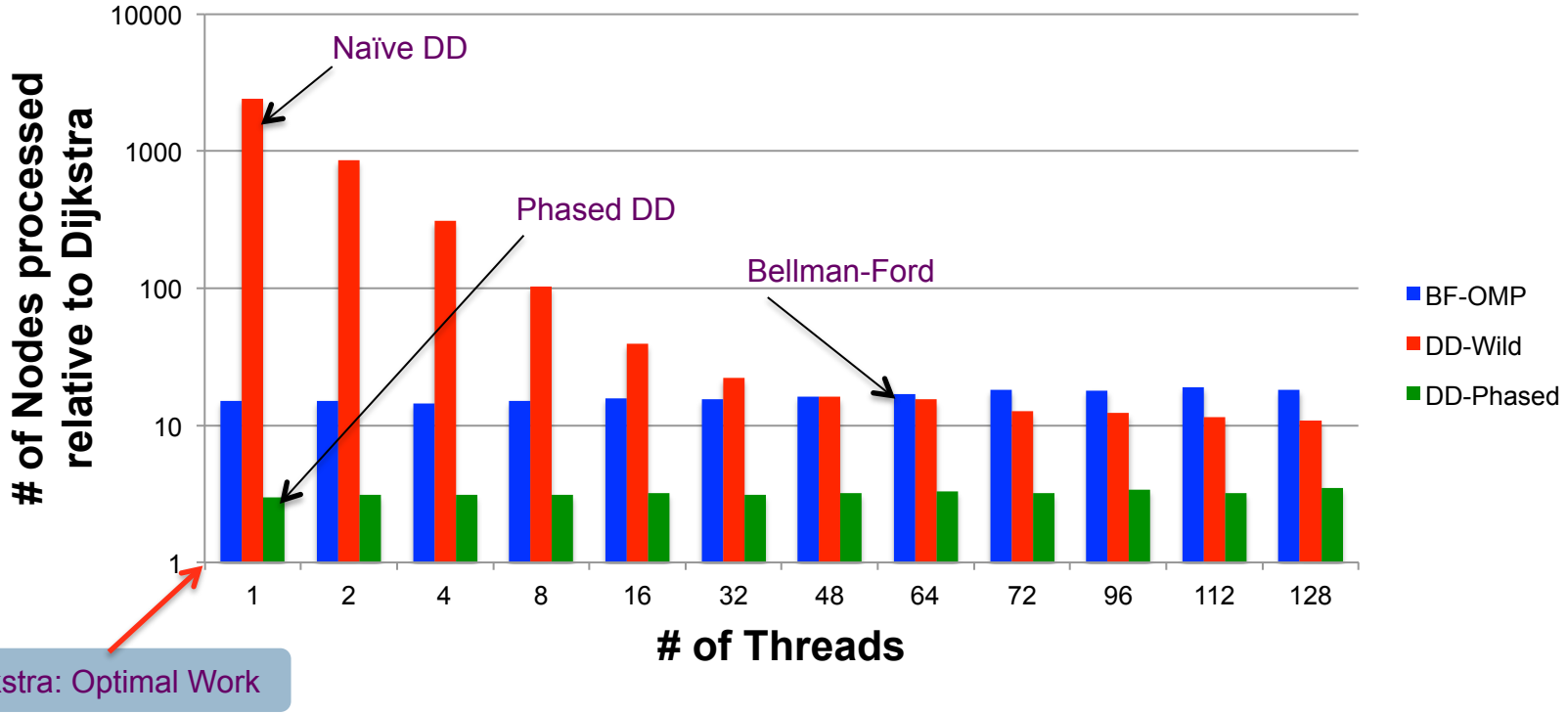
SSSP Scalability Results

Architecture:
128-thread 2-socket SPARC T2+ system

Input graph: ca-HepPh
#vertices: 12008
#edges: 237042



SSSP Work Chart



See the paper for…

- Task Groups
  - Phased Task Groups
- More detailed evaluation
  - SSSP
  - Communities
  - Betweenness Centrality

Takeaways…

Takeaways…

- Task ordering matters a lot for many data-driven computations

Takeaways…

- Task ordering matters a lot for many data-driven computations
- Need the ability to constrain task order

Takeaways…

- Task ordering matters a lot for many data-driven computations
- Need the ability to constrain task order
- ***Phased*** execution of tasks hits a sweet-spot

Takeaways…

- Task ordering matters a lot for many data-driven computations
- Need the ability to constrain task order
- ***Phased*** execution of tasks hits a sweet-spot
- Performance boost of several orders of magnitude

Takeaways…

- Task ordering matters a lot for many data-driven computations
- Need the ability to constrain task order
- ***Phased*** execution of tasks hits a sweet-spot
- Performance boost of several orders of magnitude
- More to do
  - Explore more workload classes – may lead to newer and better abstractions
  - Distributed implementation

Hardware and Software

ORACLE

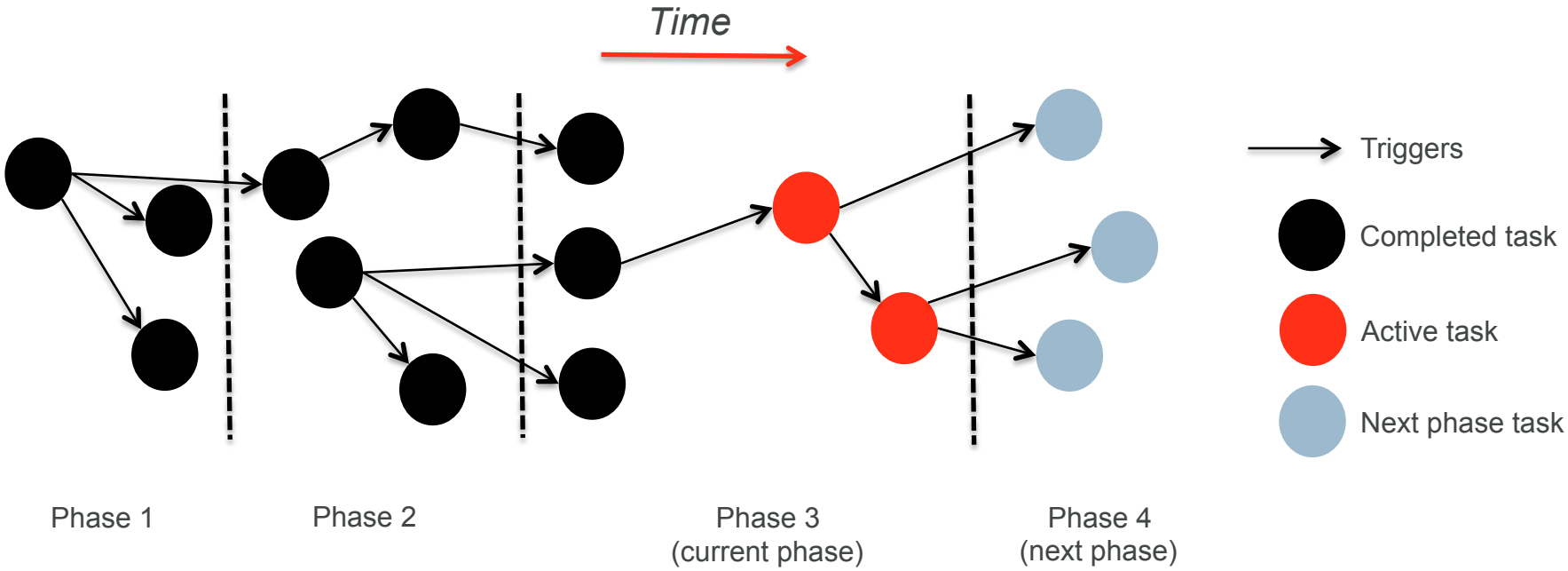
Engineered to Work Together


ORACLE®

Data-Driven Computations

- Computation driven by dynamic data dependencies
- Not new
  - Dataflow machines (since 1970s)
  - Event-driven programming – around forever (e.g. interrupt handlers, GUI, sensor networks, SEDA, etc.)
  - Database triggers
- More recently
  - Data triggered threads
  - Data-driven tasks in Habanero
  - Incremental or self-adjusting computation frameworks

Phased Execution of Tasks



Takeaways…

- Explore more abstractions
  - Go beyond having the current and next phase
  - DAG
  - Provide primitives to express ordering between task groups (like Dryad, Hive, etc.)
- Go distributed