DARE: Making Diffusing Computations More Efficient for Loop-Free Shortest-Path Routing

J.J. Garcia-Luna-Aceves

Morteza Moghaddassian

University of Toronto

{jj.garcialunaaceve,m.moghaddassian}@utoronto.ca

ACM/IRTF ANRW 2024

UNIVERSITY OF TORONTO

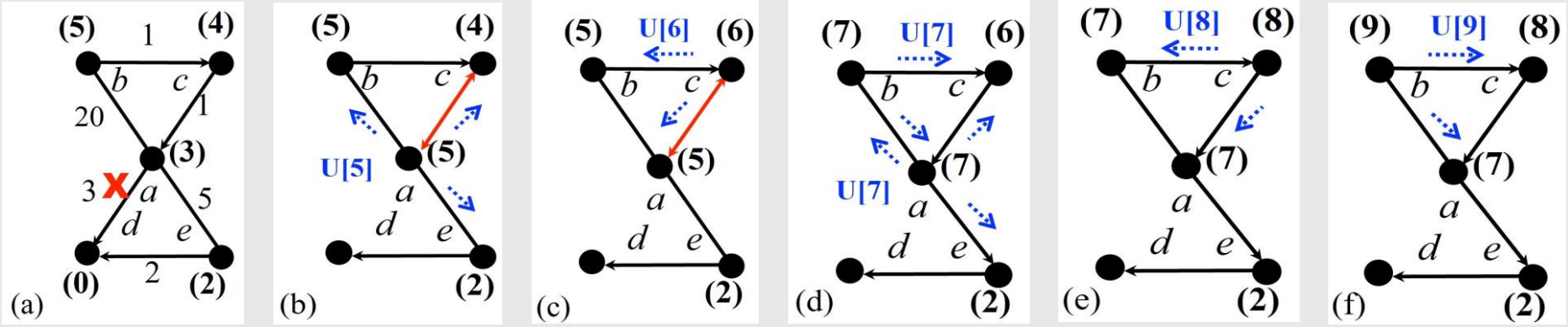
Outline

- **Motivation:** Why do we need yet another routing algorithm for diffusing computation?
- **DARE:** Faster diffusing computation with less signaling
- **Comparison** of DARE and DUAL routing methods

UNIVERSITY OF TORONTO

Distributed Bellman-Ford Algorithm (DBF)

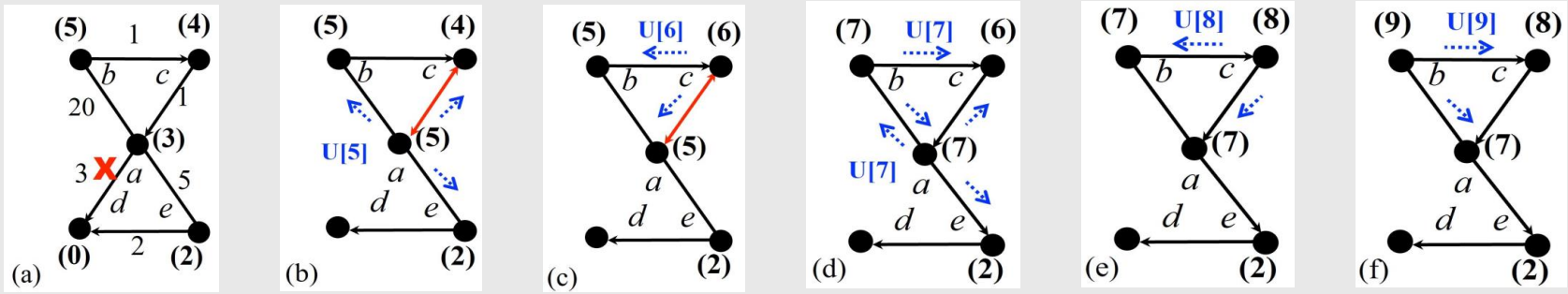
- DBF works **very efficient for minimum-hop routing** in many scenarios.
- DBF **main issue** is **count-to-infinity** after destination failure or network partitioning.
- DBF may **cause short or long-lived temporary routing loops** after link failures happens or the link-cost increases.



UNIVERSITY OF TORONTO

Distributed Bellman-Ford Algorithm (DBF)

- DBF works **very efficient for minimum-hop routing** in many scenarios.
- DBF **main issue** is **count-to-infinity** after destination failure or network partitioning.
- DBF may **cause short or long-lived temporary routing loops** after link failures happens or the link-cost increases.



Only a few steps in for convergence but temporary loops emerged.

Diffusing Update Algorithm (DUAL)

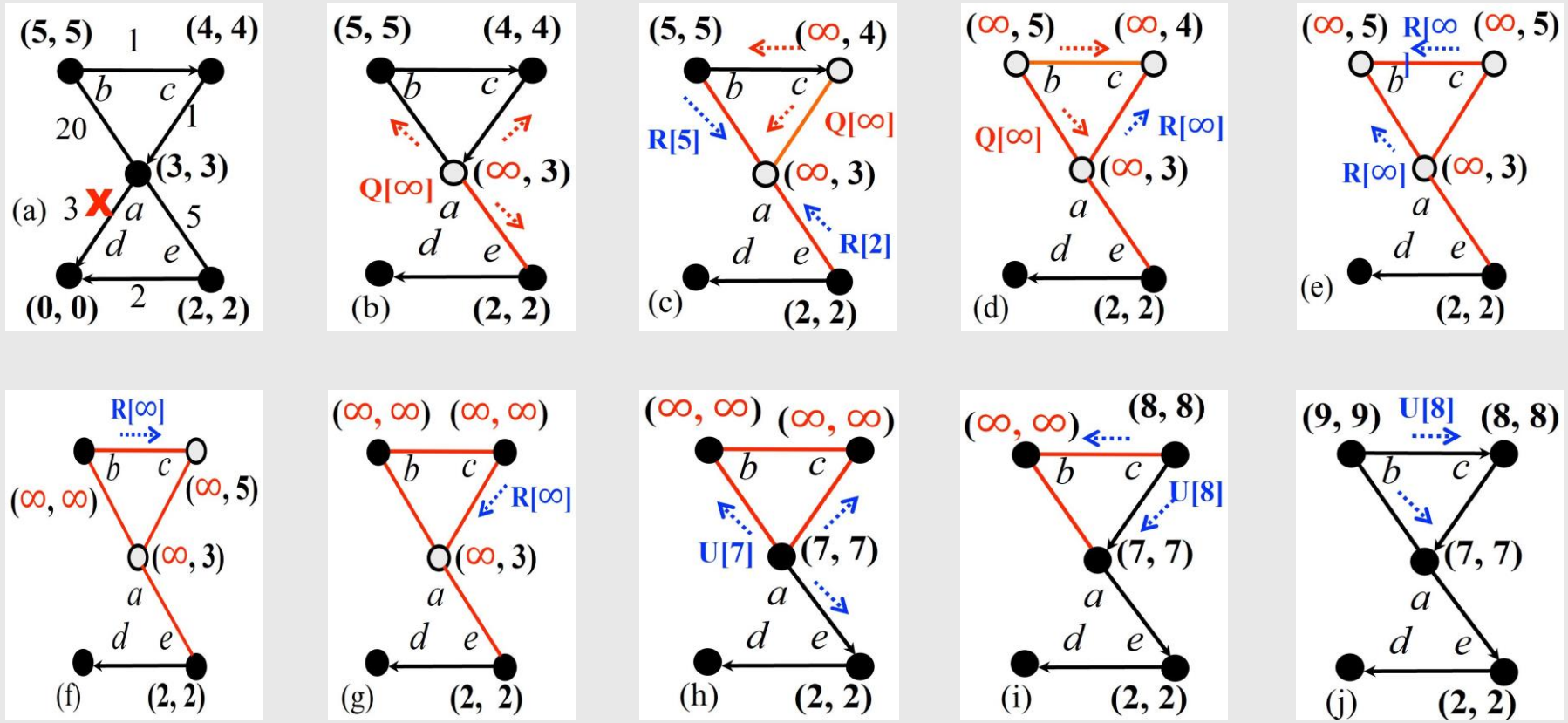
- Using **query-reply signaling** based on feasible distances to eliminate routing loops.
- Condition for loop-free next hops selection

$$\mathcal{SNC}: (h_d^q < f_d^a) \wedge (h_d^q = Min\{h_d^n + l(a,n) | n \in N^a\})$$

- If SNC is not satisfied by any neighbor:
  - Set $\boldsymbol{f_d^a = h_\infty}$ & **query** all neighbors stating **new distance for $\boldsymbol{h_d^a}$ (Diffuse)**
  - **Wait** for all neighbors to reply (**Blocking in Active mode**).
  - Compute a new distance → SNC satisfied → Go to Passive mode.

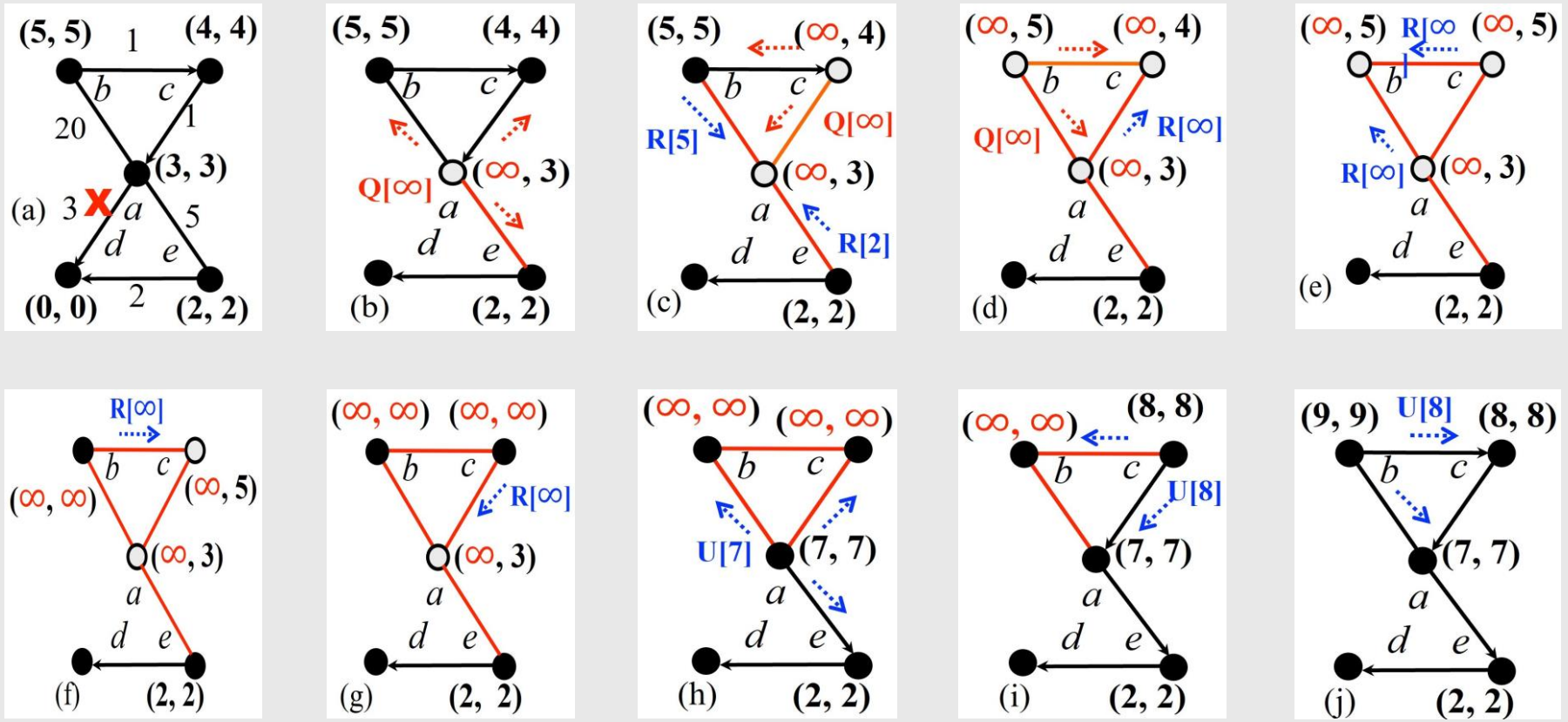
UNIVERSITY OF TORONTO

Example of Loop-Free Routing in DUAL



UNIVERSITY OF TORONTO

Example of Loop-Free Routing in DUAL



More steps, Several Blocking Routers, More Signaling, but no Routing Loops occur.

UNIVERSITY OF TORONTO

Key Issues with DUAL

- DUAL was meant to provide only a single loop-free next hop per destination at each router.
  - Having multiple next hop choices can reduce signaling.

- A DUAL router that sends a query (**diffuse**) **cannot** use a new next hop until it receives reply from all its neighbors.

  - Some neighbors may offer loop-free routers that are **valid**.
  - **Unnecessary blocking** while diffusing can block the flow of data packets in the data plane.

UNIVERSITY OF TORONTO

The Road to DARE!

- DARE is **compatible**! → only changing the meaning of **feasible next hop** (successor) as used in DUAL.
  - **feasible successor in DUAL**: a neighbor router reporting a distance value **smaller than** a **feasible distance value** & be **the shortest distance** available.
  - **feasible successor in DARE**: a neighbor router reporting a **smaller distance** than the **last distance** the router had **at the time it was passive**.
    - Use neighbor routers that reported **shorter distances** than the distance the **router have reported to its neighbors** → Call these routers **the ordered routers.**
    - **Diffuse** only if **no ordered router is available** or **none** of the ordered neighbors offer shorter distances.
    - **Decouple** shortest path calculation from the selection of successor neighbors.

UNIVERSITY OF TORONTO

The Road to DARE - Continued

- DARE **remembers** what DUAL **forgets**!
  - **DUAL** routers **do not remember** the neighbor that forced them to go into a diffusing computation.
    - Send replies to **every** neighbor! → **Much** signaling overhead.
    - Can only participate in **a single diffusing computation** at any given time.
  - **DARE** routers **remember** the neighbor that forced them to go into a diffusing computation.
    - Reply **only** to the neighbor remembered! → **Less** signaling overhead.
    - Can participate in **multiple diffusing computation**! Active nodes can **merge** diffusing computations.

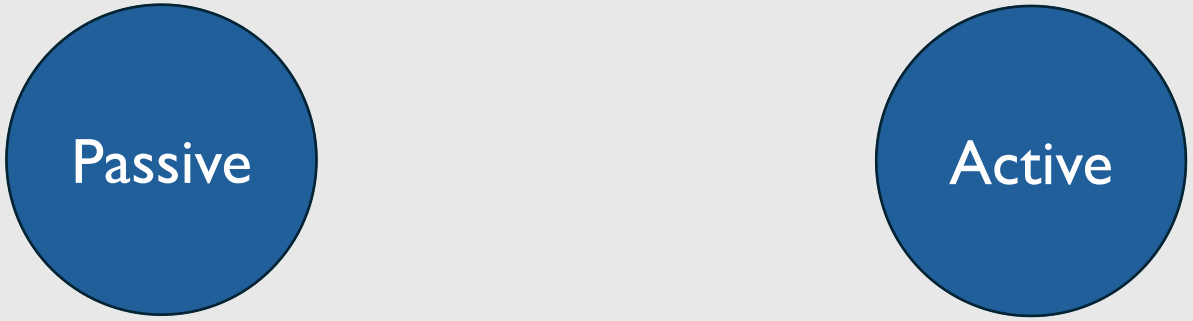
UNIVERSITY OF TORONTO

The Road to DARE - Continued

- DARE is **fair**!
  - **DUAL** routers can **only** change successor when a **local condition is satisfied**, or a **diffusing computation completes**.
  - **DARE** routers can change successor **at any time** only if the successor neighbor reports a shorter distance held by the router when it was passive.
  - **DUAL** routers use a complex state machine to account for multiple diffusing computations.
    - Hard to evolve and include multiple performance criteria.
  - **DARE** routers use a much simpler state machine.
    - Much more flexibility for using more performance criteria.

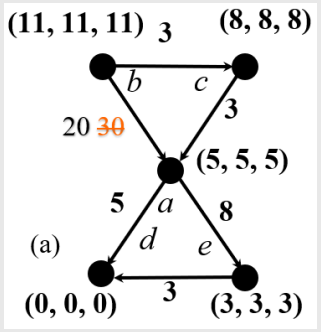
UNIVERSITY OF TORONTO

The DARE State Machine



UNIVERSITY OF TORONTO

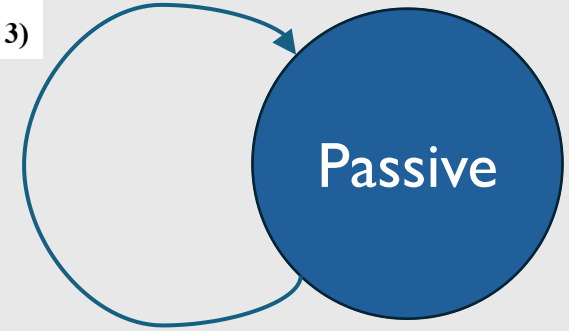
The DARE State Machine



$(link\ change/update)$

$\wedge$

$LDC = T$



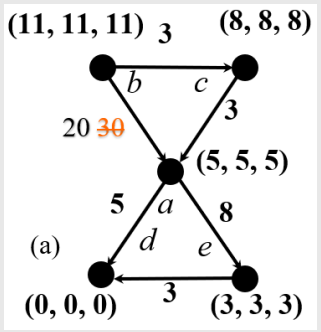
$Q[RD_d^v, v \in N^k]$

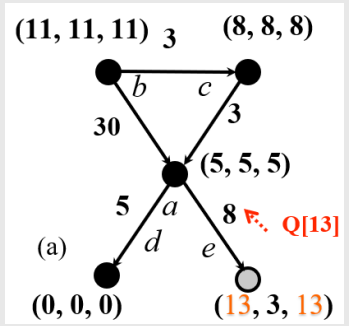
$\wedge$

$LDC = T$

UNIVERSITY OF TORONTO

The DARE State Machine





$(link\ change/update)$

$\wedge$

$LDC = T$

$(input\ event) \wedge LDC = F$

Passive

Active

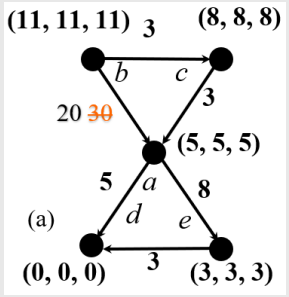
$Q[RD_d^v, v \in N^k]$

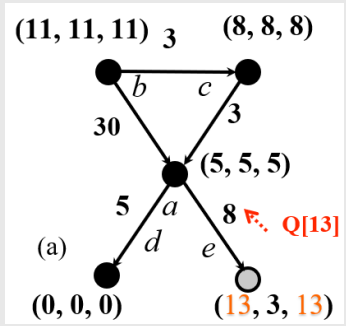
$\wedge$

$LDC = T$

UNIVERSITY OF TORONTO

The DARE State Machine





$(link\ change/update)$

$\wedge$

$LDC = T$

$(input\ event) \wedge LDC = F$

$[\forall\, v \in N^k (C_{dv}^k = T)]$

$\wedge$

$(input\ event)$

Passive

Active

$Q[RD_d^v, v \in N^k]$

$\wedge$

$LDC = T$

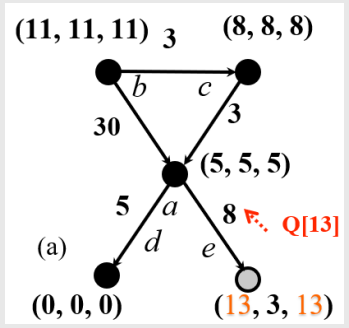
$[\forall\, v \in N^k (C_{dv}^k = F)]$

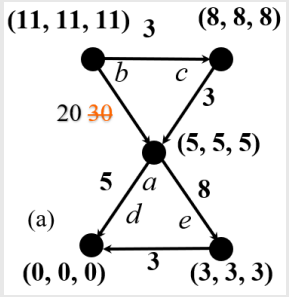
$\wedge$

$LDC = F$

UNIVERSITY OF TORONTO

The DARE State Machine





$(link\ change/update)$
$\wedge$
$LDC = T$

$(input\ event) \wedge LDC = F$

$\left[\forall\, v \in N^k (C_{dv}^k = T)\right]$
$\wedge$
$(input\ event)$

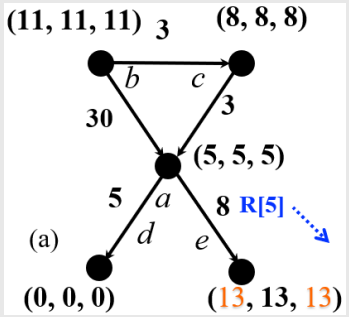
Passive

Active

$\left[\forall\, v \in N^k (C_{dv}^k = F)\right] \wedge (LDC = T)$

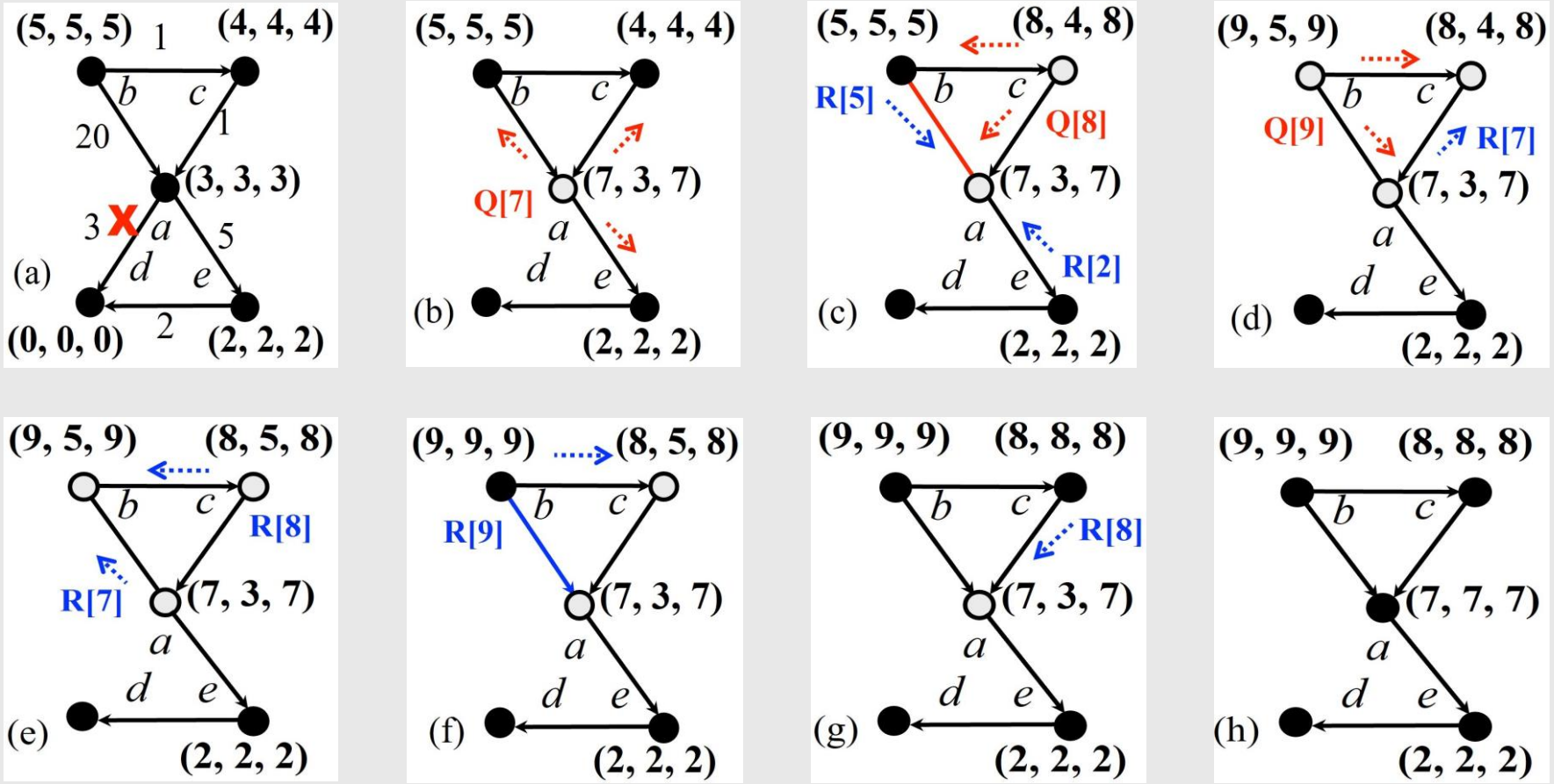
$Q[RD_d^v, v \in N^k]$
$\wedge$
$LDC = T$

$\left[\forall\, v \in N^k (C_{dv}^k = F)\right]$
$\wedge$
$LDC = F$



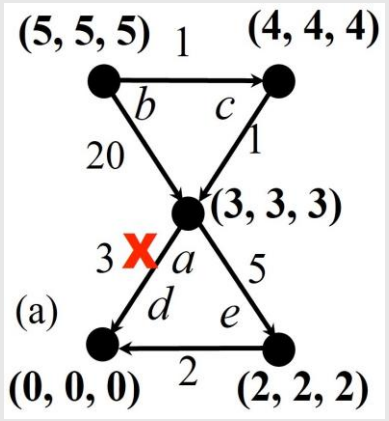
UNIVERSITY OF TORONTO

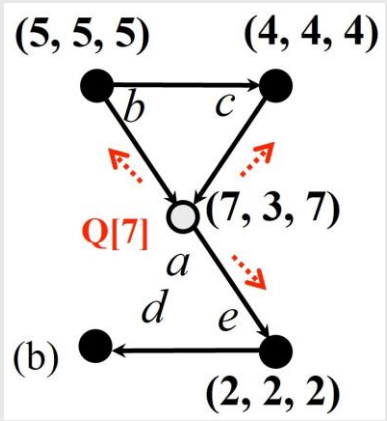
Example: Link Failure in DARE

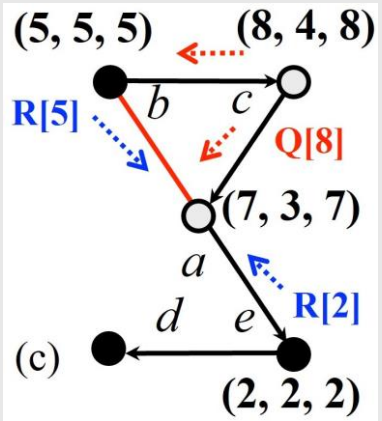


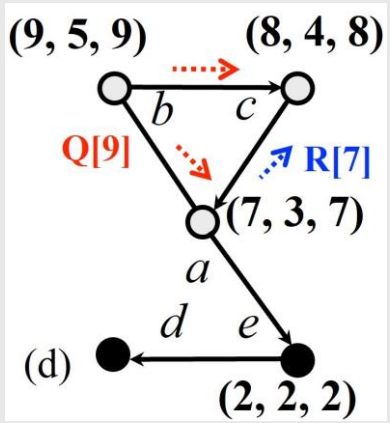
UNIVERSITY OF TORONTO

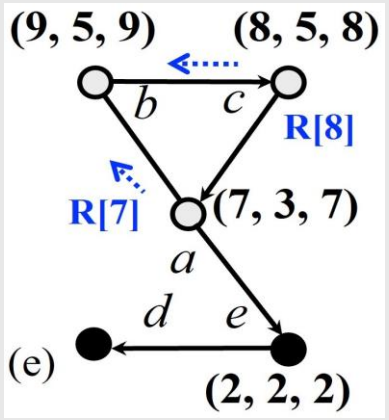
Example: Link Failure in DARE

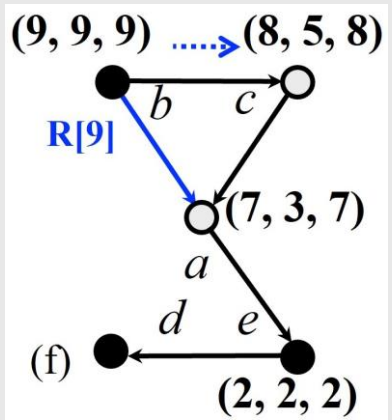


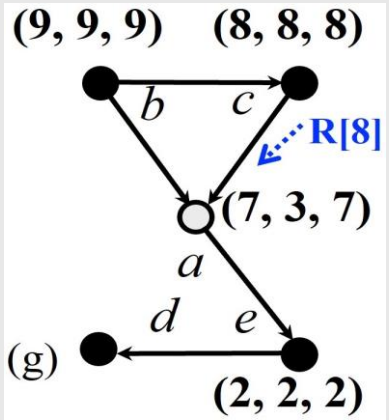


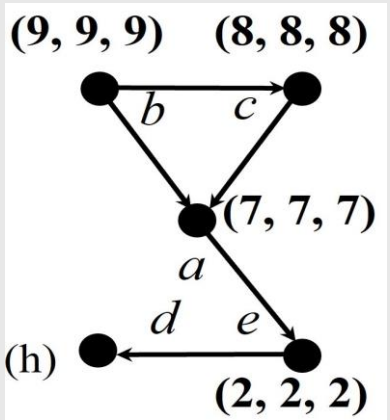












Fewer steps than DUAL, less overhead, and no routing loop occurs!

UNIVERSITY OF TORONTO

Conclusion

- **DARE** is a new routing algorithm using **diffusing computation**.
- DARE speaks the same language as DUAL does but DARE:
  - is more efficient and uses a **simpler state machine**.
  - uses **less signaling** overhead for synchronization.
  - can switch successor nodes **at any time**.
  - Can **converges faster**!
- A step forward is to use DARE in a routing protocol.

UNIVERSITY OF TORONTO

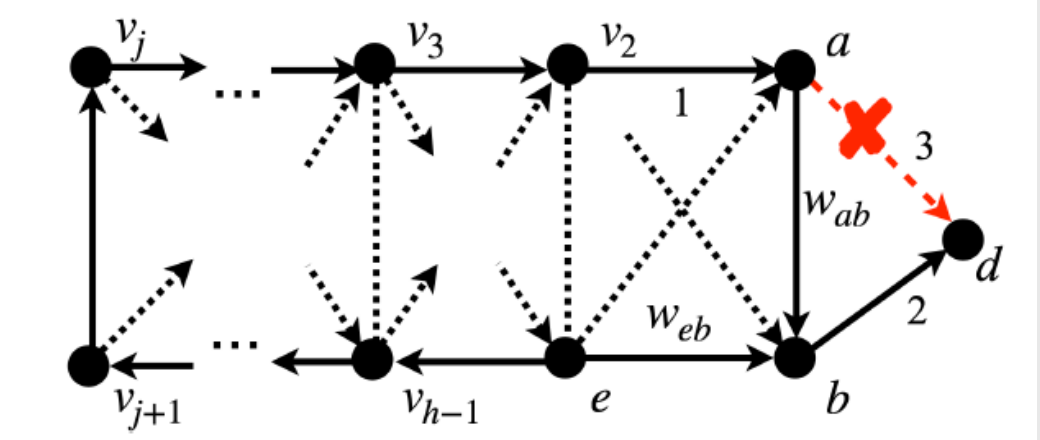
Thank you for your time!

Questions

UNIVERSITY OF TORONTO

Q/A: Results (Single Node Convergence)

Network Size	Algorithm	Total Messages	Blocking Steps at a
10	DUAL	30	14
	DARE	16	0
30	DUAL	110	54
	DARE	56	0
50	DUAL	190	94
	DARE	96	0



Less messaging & Zero blocking!

UNIVERSITY OF TORONTO