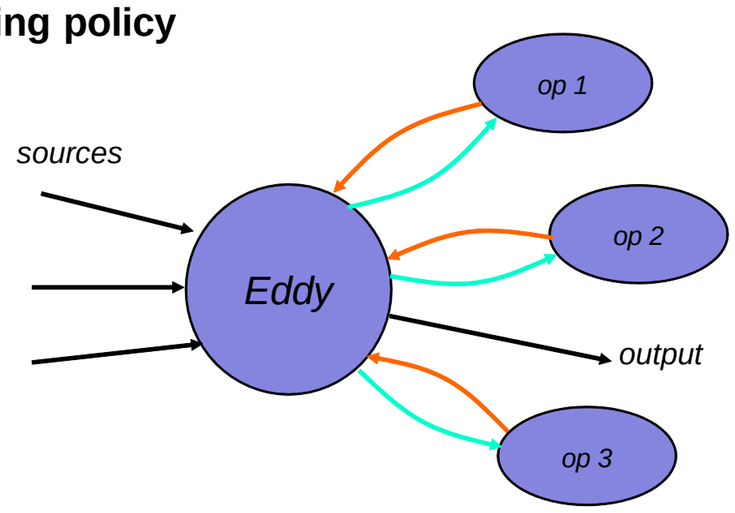
A Reinforcement Learning Approach for Adaptive Query Processing

Kostas Tzoumas, Timos Sellis, Christian S. Jensen

AQP with eddies

- Decides an operator to *route* an incoming tuple
- Uses additional meta-data for correctness (tuple descriptor)
- Uses a **routing policy**



Problem statement

- The routing policy component
  - random, lottery scheduling, rank ordering, ...
- How does query optimization enter the picture?

- Beginning with zero knowledge about the environment and the data
- Optimization
  - find an optimal query plan
- Adaptation
  - change it when the environment changes
- It is a learning problem

Reinforcement Learning

receive state s_t
choose an action α_t
execute α_t
receive next state s_{t+1}
receive reward r_{t+1}
update Q values

the Q values $Q(s,\alpha)$

state space S, action spaces A(s)
α = chooseAction (Q, s)
reward function r (s, α)
Q (s, α) = update (Q, r, s')

Choosing and updating

α = chooseAction(Q,s)

- random
  - choose a random action
- greedy
  - choose the action with the highest *Q(s,α)*
- ε-greedy
  - greedy with prob. *1-ε*
  - random with prob. *ε*
- uniform
  - prob. to choose *α* propotional to *Q(s,α)*
- simulated annealing
  - prob. propotional to *exp (Q(s,α) / T)*

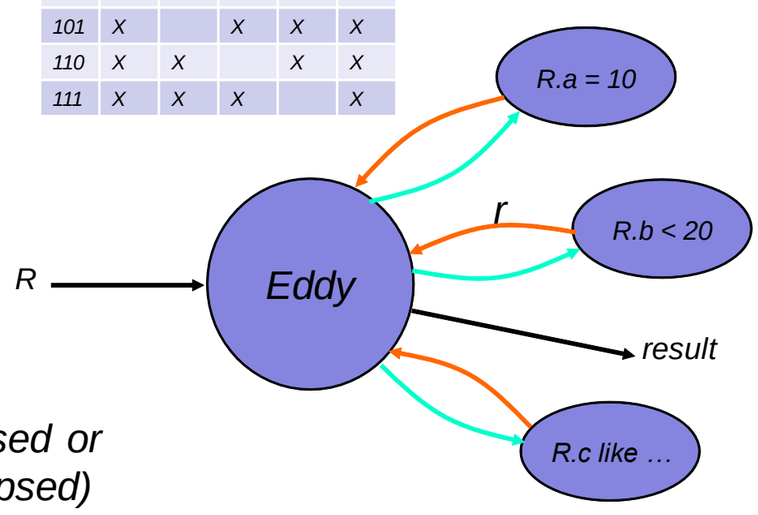
Q(s,α) = update (Q, r, s')

- Monte Carlo
  - average of rewards
  - *Q = Q + 1/n (r - Q)*
- constant-α MC
  - weighted average
  - *Q = Q + α (r - Q)*
- Q
  - bootstrapping
  - *Q = Q + α [r + γ maxQ(s',α') - Q]*

Selections

receive state *s=000*
A(s) = {σ1, σ2, σ3}
choose action *α=σ2*
send tuple to *σ2*
wait for *σ2* to return next state *s'=010*
receive reward *r*
update Q values

Q	σ1	σ2	σ3	out	get
null	X	X	X	X	
000				X	X
001			X	X	X
010		X		X	X
011		X	X	X	X
100	X			X	X
101	X		X	X	X
110	X	X		X	X
111	X	X	X		X

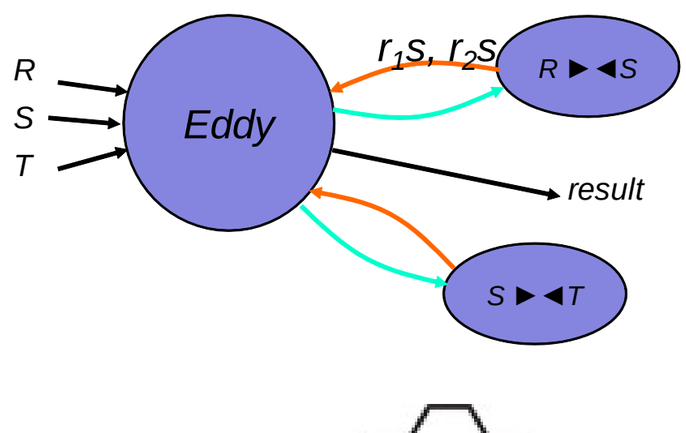


r = (20, 13, "abc")
reward = -1 or 0
reward= - time elapsed or
- ε(time elapsed)

Binary Symmetric Hash Joins

receive state *s=S*
A(s)={R ►◄S,S ►◄T}
choose action *α=R ►◄S*
route tuple and push *R ►◄S* in a stack
pop intermediate tuple
next state *s'=RS*
receive reward *r*
update *Q* values
s=s'
repeat

Q	►◄1	►◄2	out	get R	get S	get T
null	X	X	X			
R		X	X	X	X	X
S			X	X	X	X
T	X		X	X	X	X
RS	X		X	X	X	X
ST		X	X	X	X	X
RST	X	X		X	X	X



AALBORG UNIVERSITY