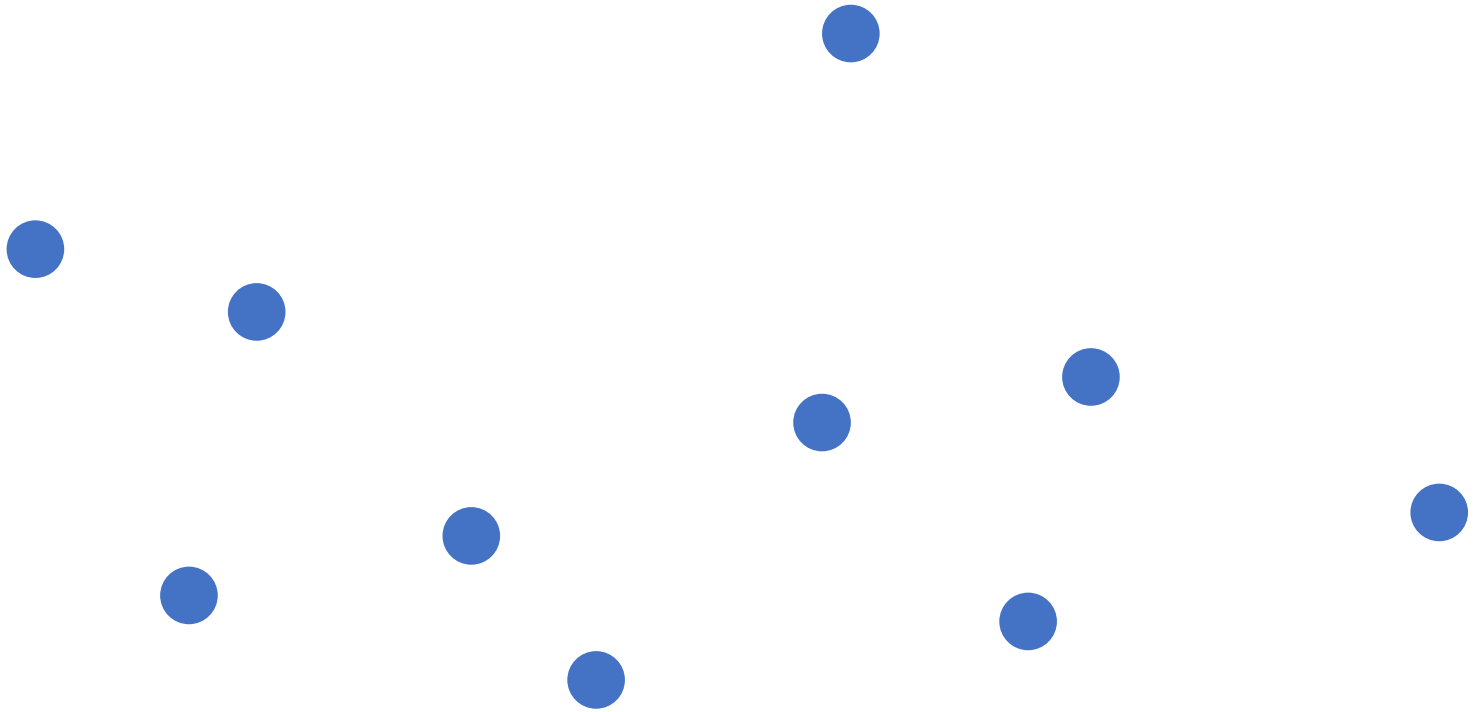
Neural Datalog Through Time

Hongyuan Mei[1], Guanghui Qin[1], Minjie Xu[2], Jason Eisner[1]

[1]Johns Hopkins University

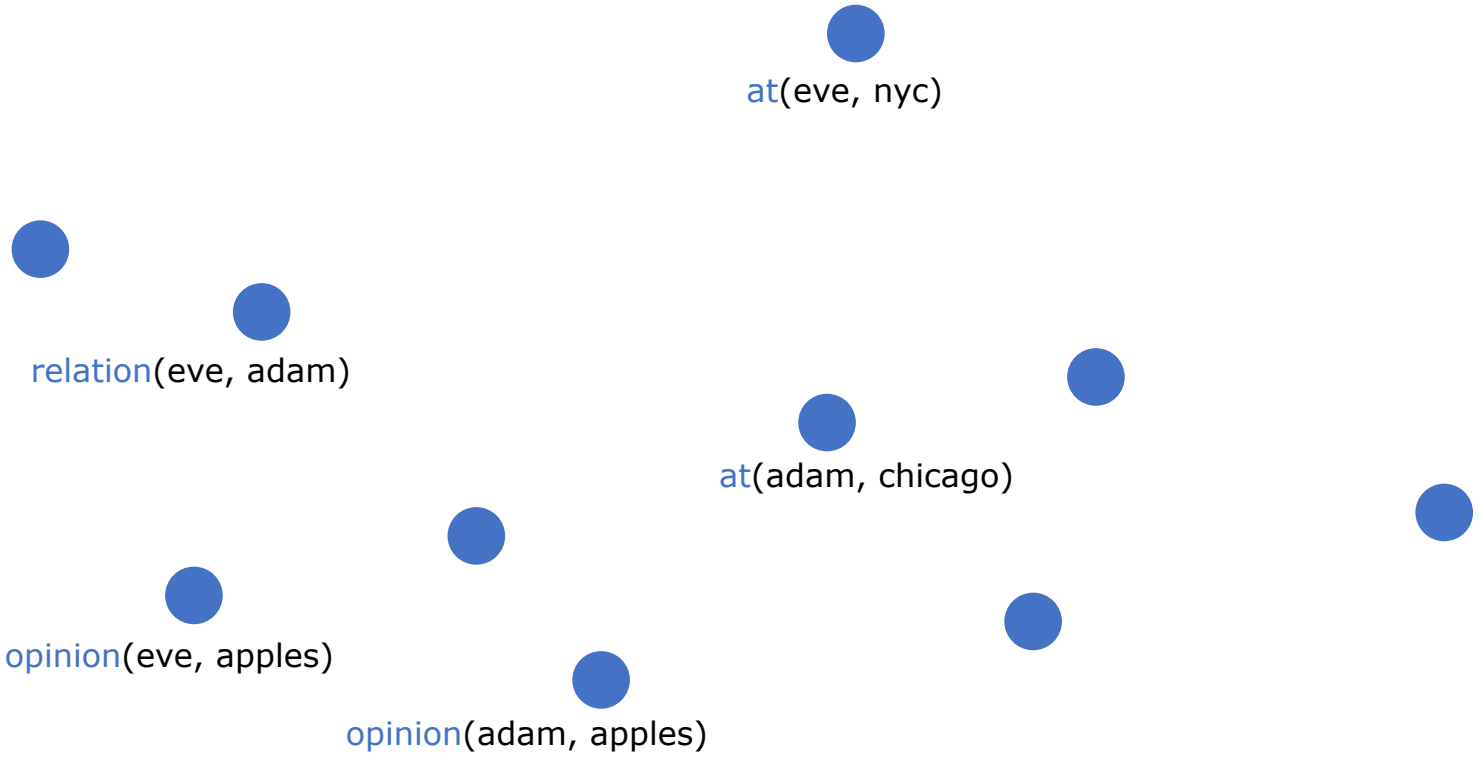
[2]Bloomberg

Model How a Database Changes Over Time



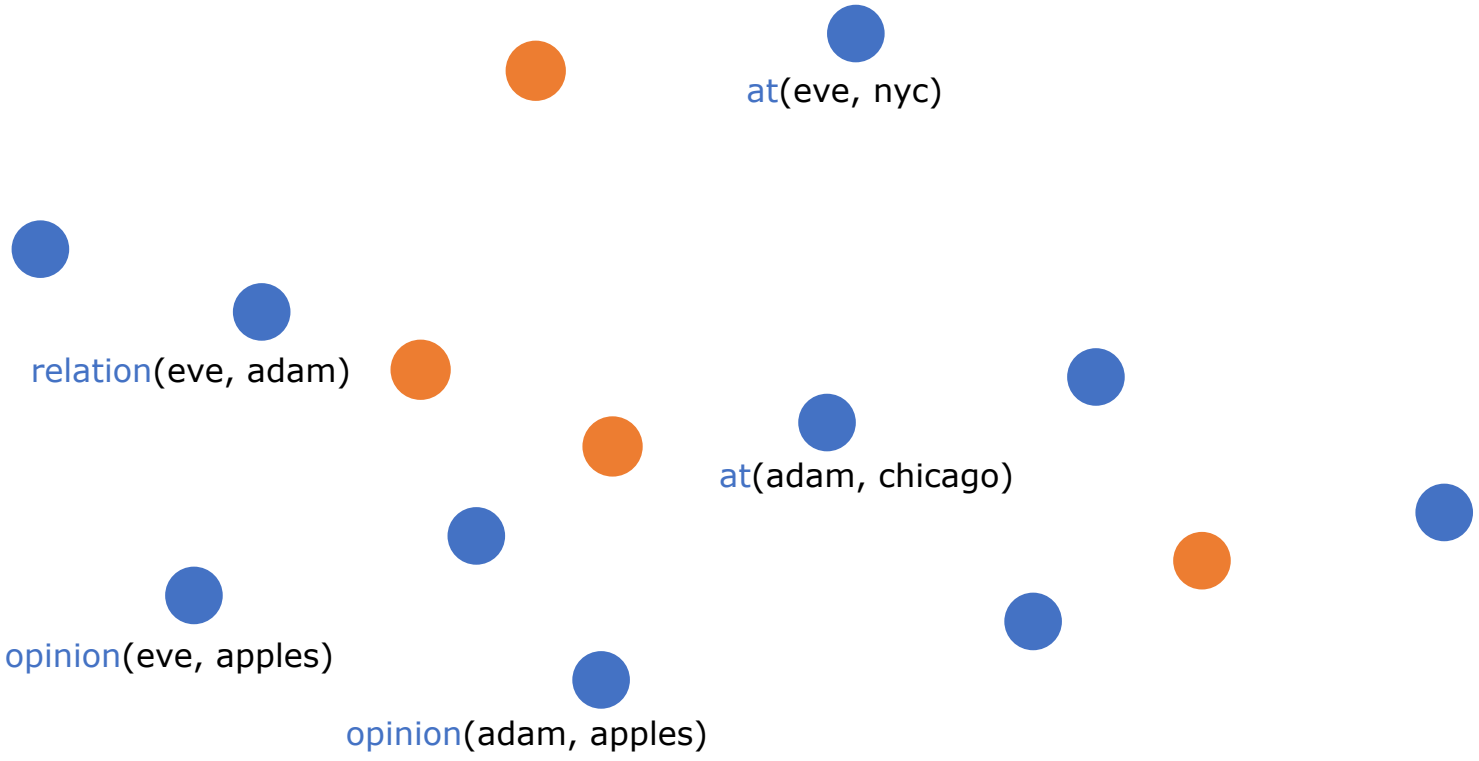
Model How a Database Changes Over Time

200,000 facts right now



Model How a Database Changes Over Time

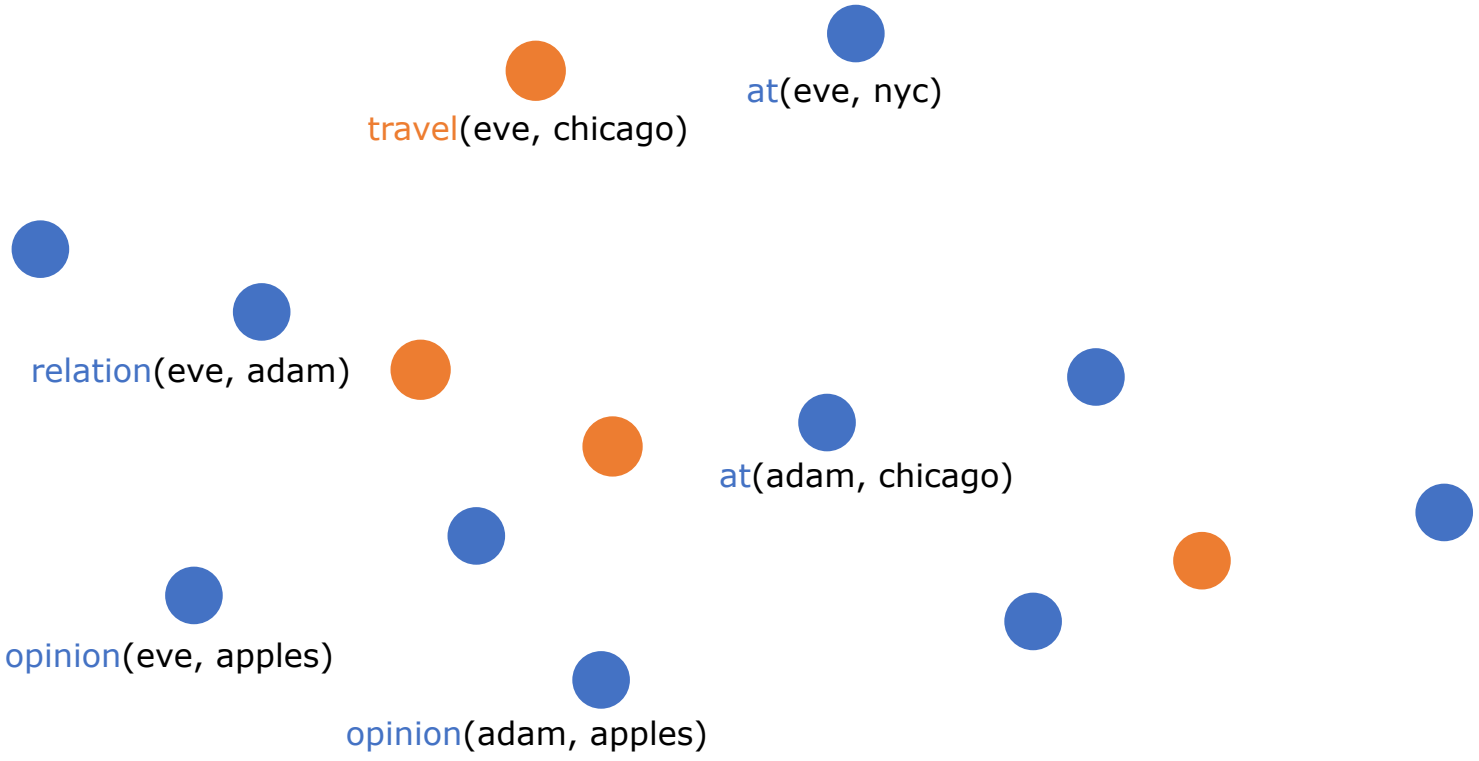
200,000 facts right now



Model How a Database Changes Over Time

200,000 facts right now

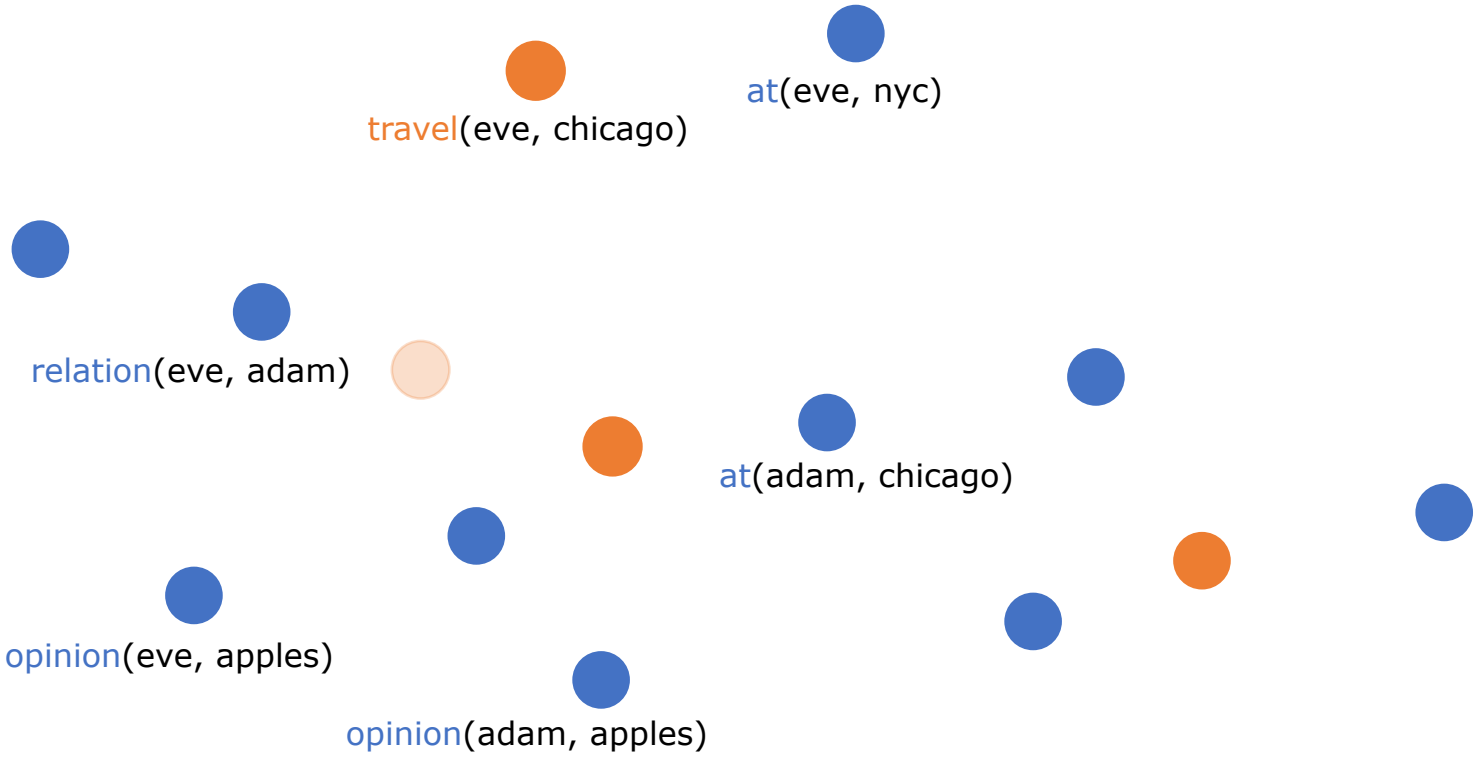
50,000 possible events right now



Model How a Database Changes Over Time

200,000 facts right now

50,000 possible events right now

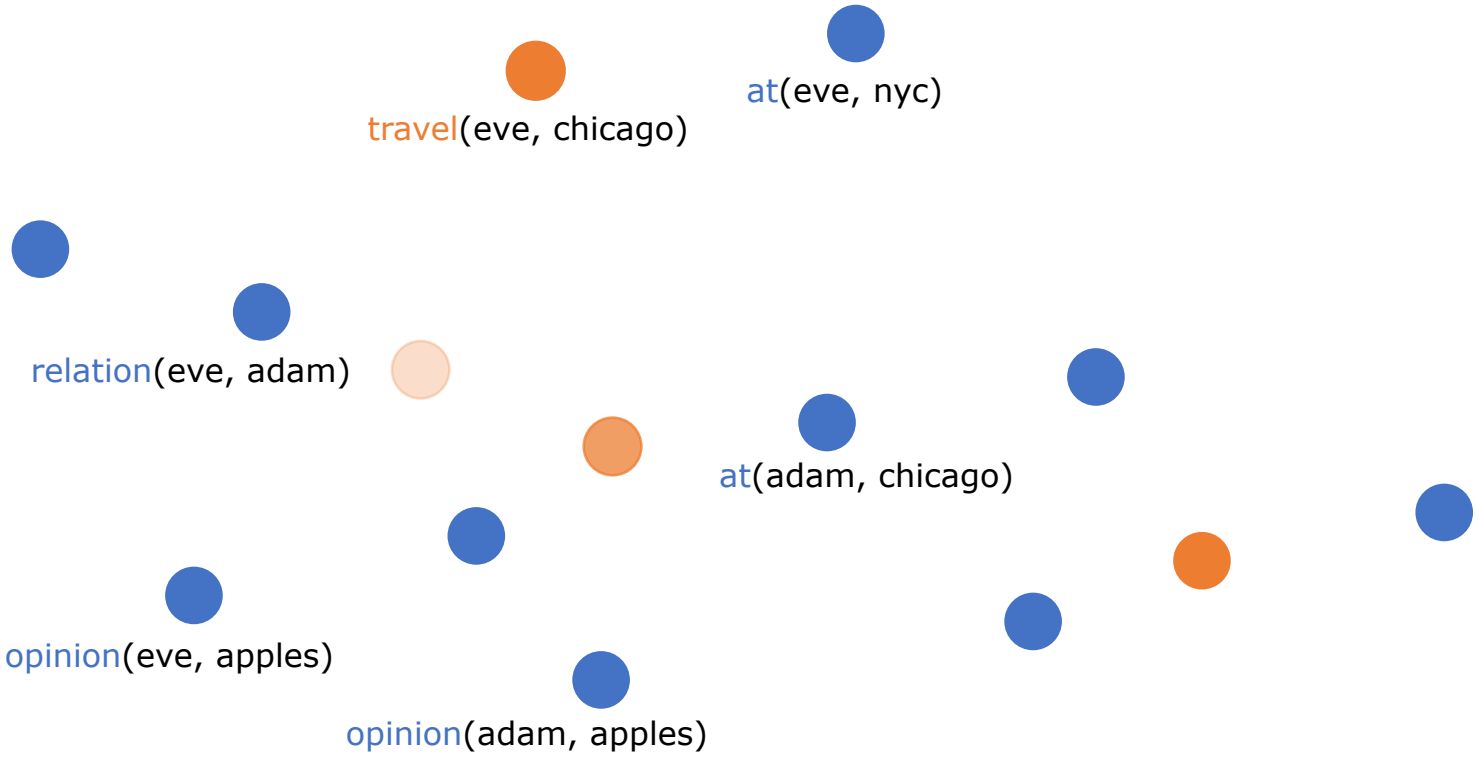


Model How a Database Changes Over Time

200,000 facts right now

50,000 possible events right now

little language
to specify a generative model
of event sequences

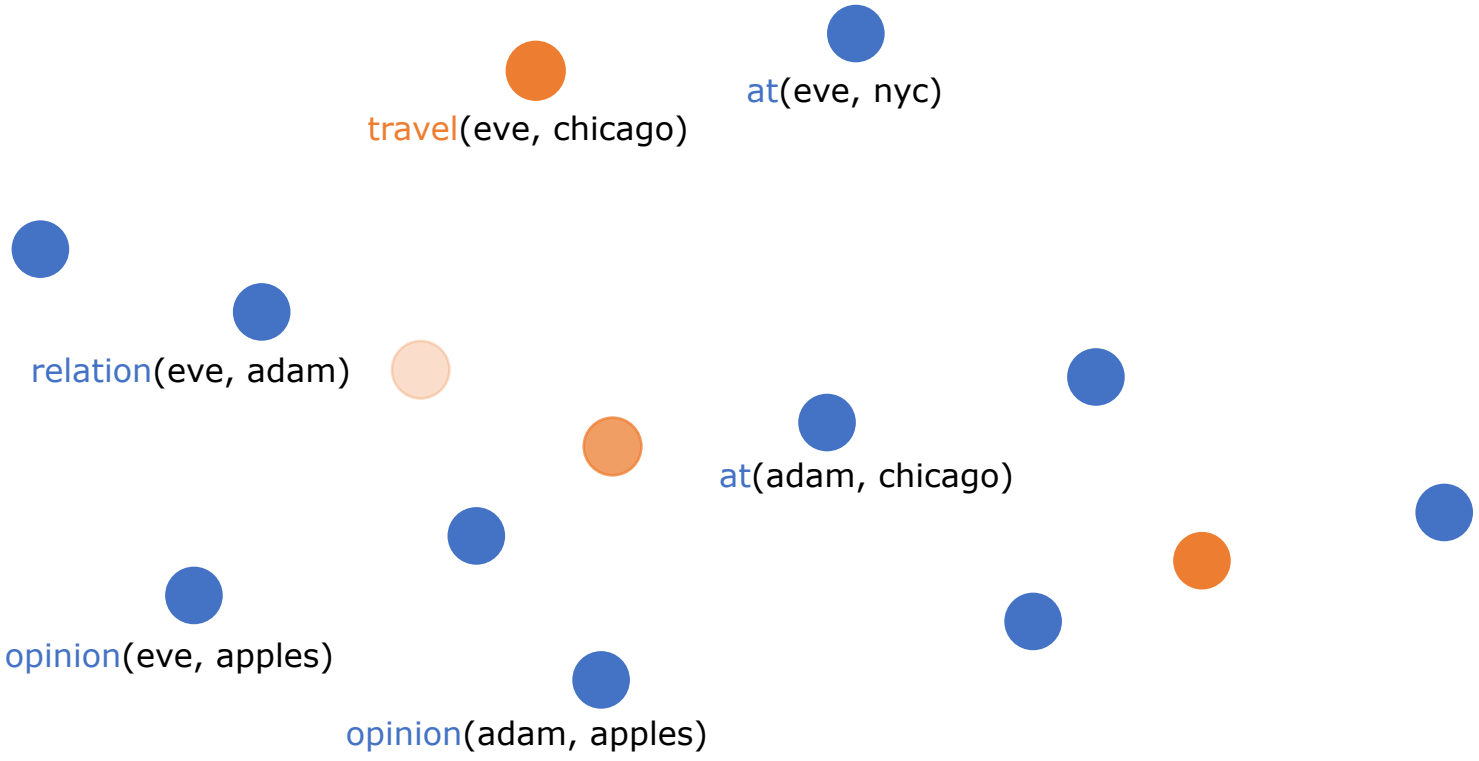


Model How a Database Changes Over Time

200,000 facts right now

50,000 possible events right now

little language
to specify a generative model
of event sequences

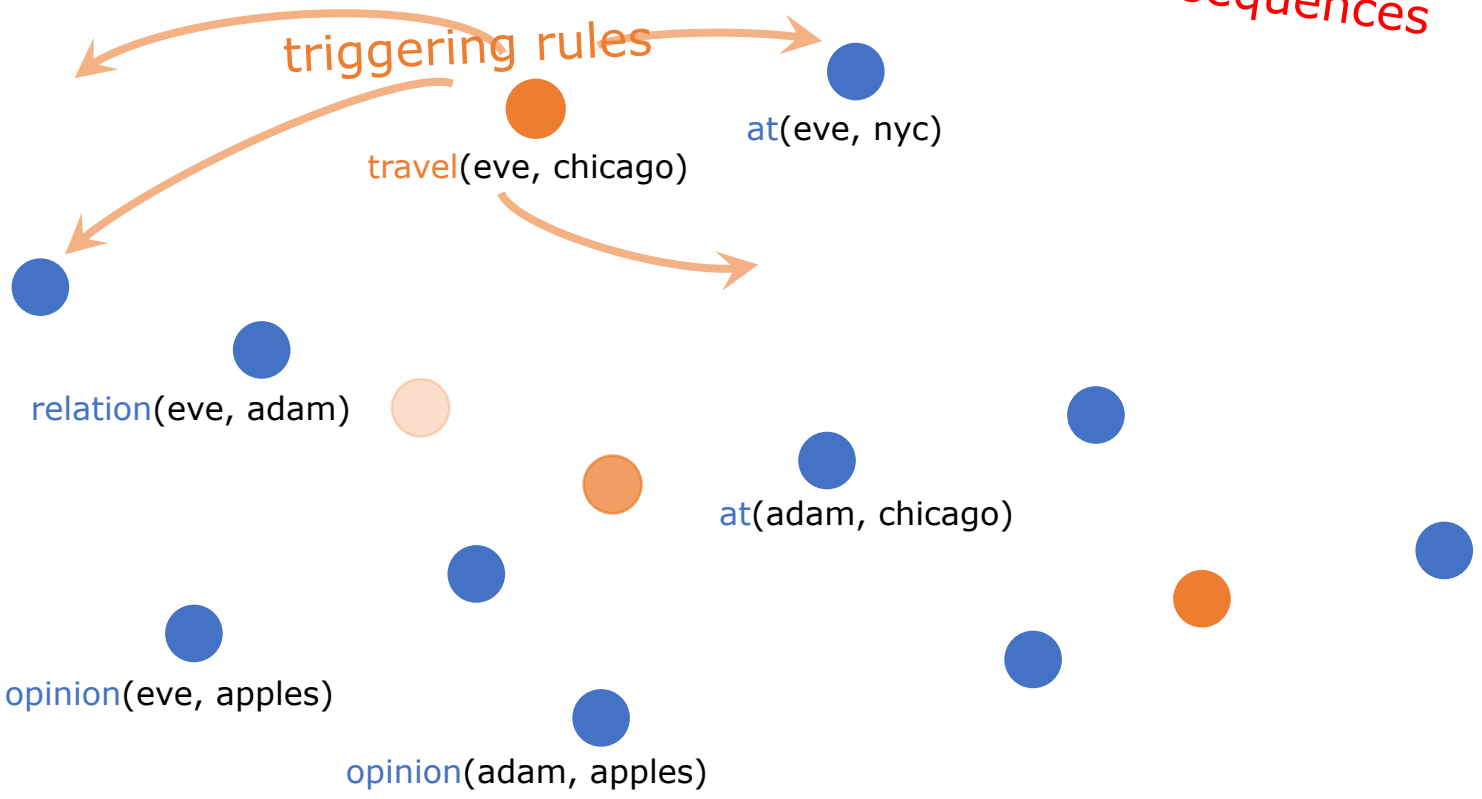


Model How a Database Changes Over Time

200,000 facts right now

50,000 possible events right now

little language
to specify a generative model
of event sequences

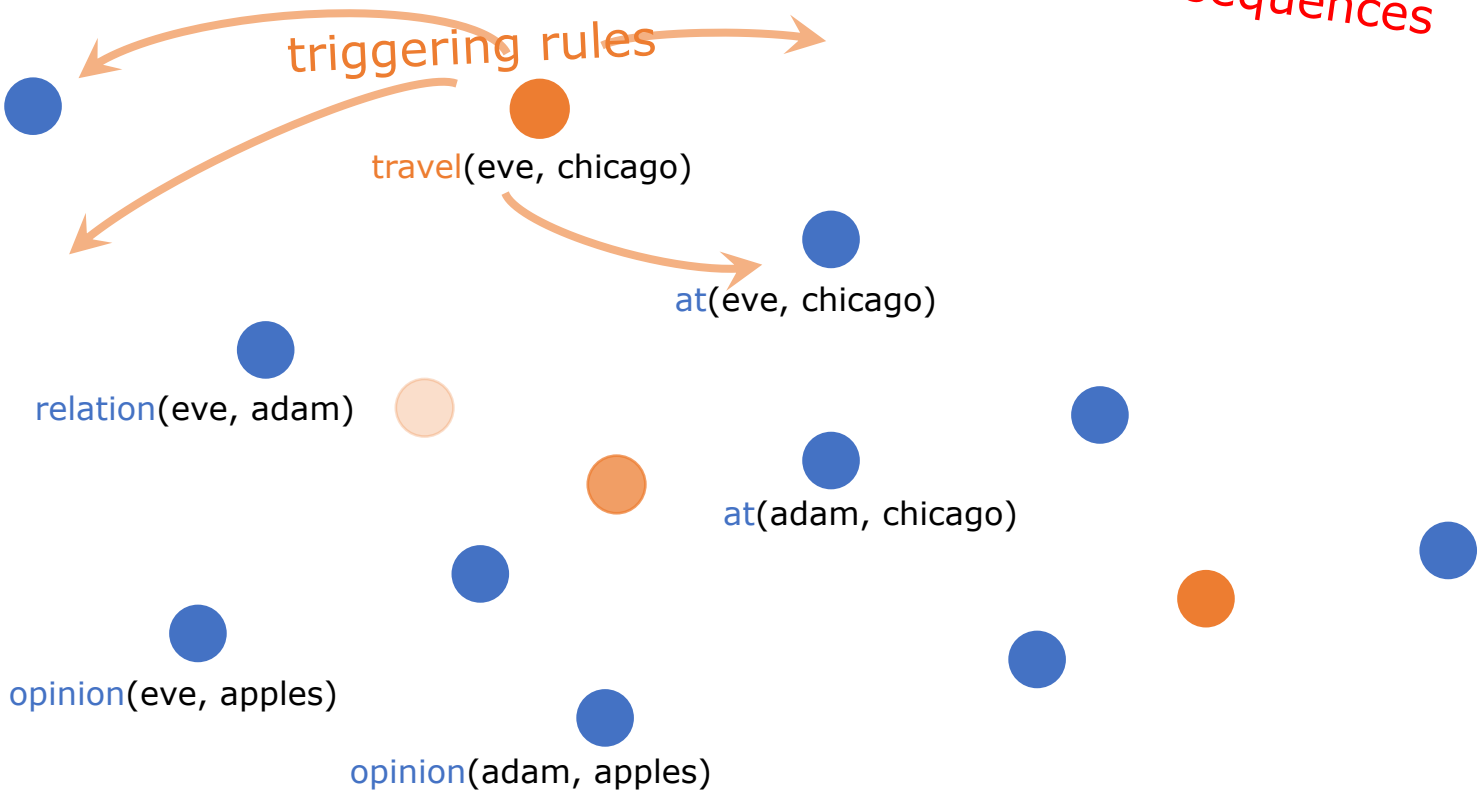


Model How a Database Changes Over Time

200,000 facts right now

50,000 possible events right now

little language to specify a generative model of event sequences

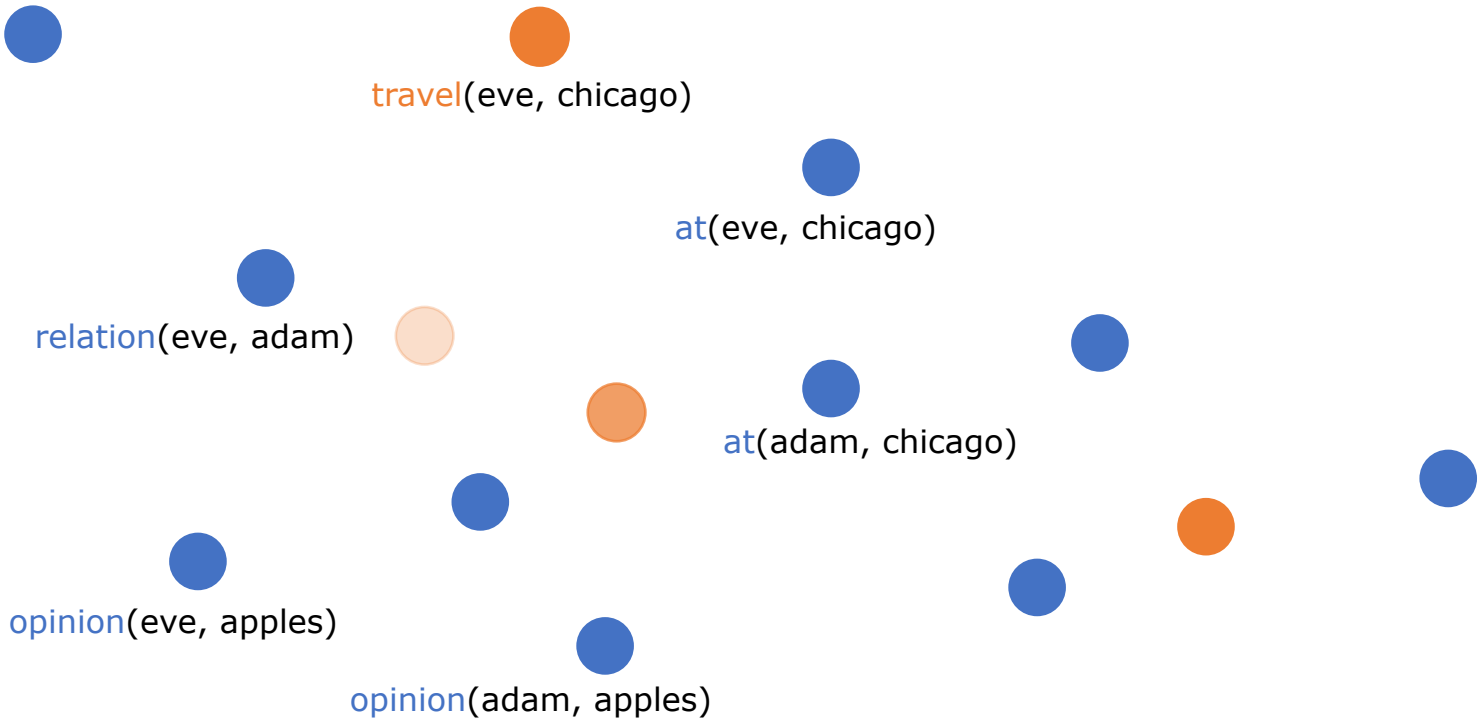


Model How a Database Changes Over Time

200,000 facts right now

50,000 possible events right now

little language to specify a generative model of event sequences

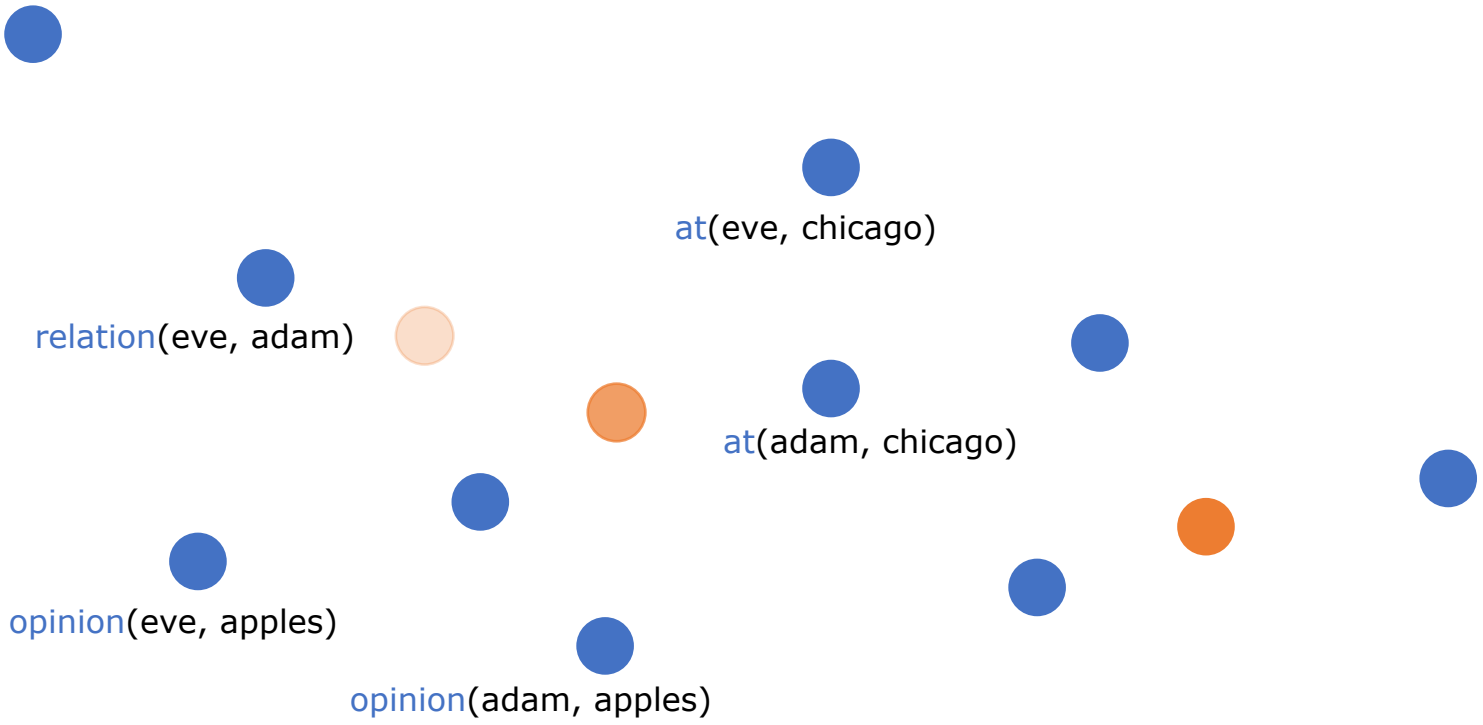


Model How a Database Changes Over Time

200,000 facts right now

50,000 possible events right now

little language to specify a generative model of event sequences

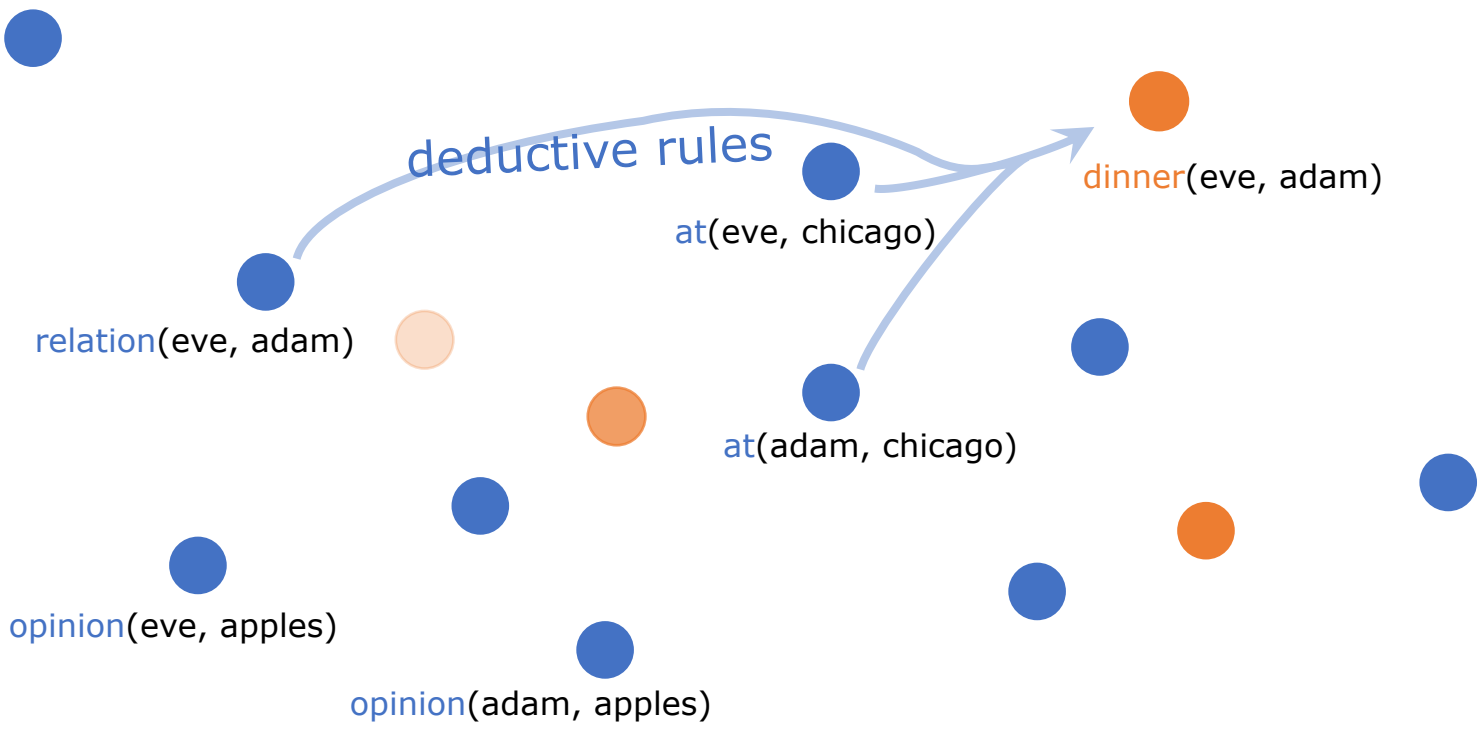


Model How a Database Changes Over Time

200,000 facts right now

50,000 possible events right now

little language to specify a generative model of event sequences

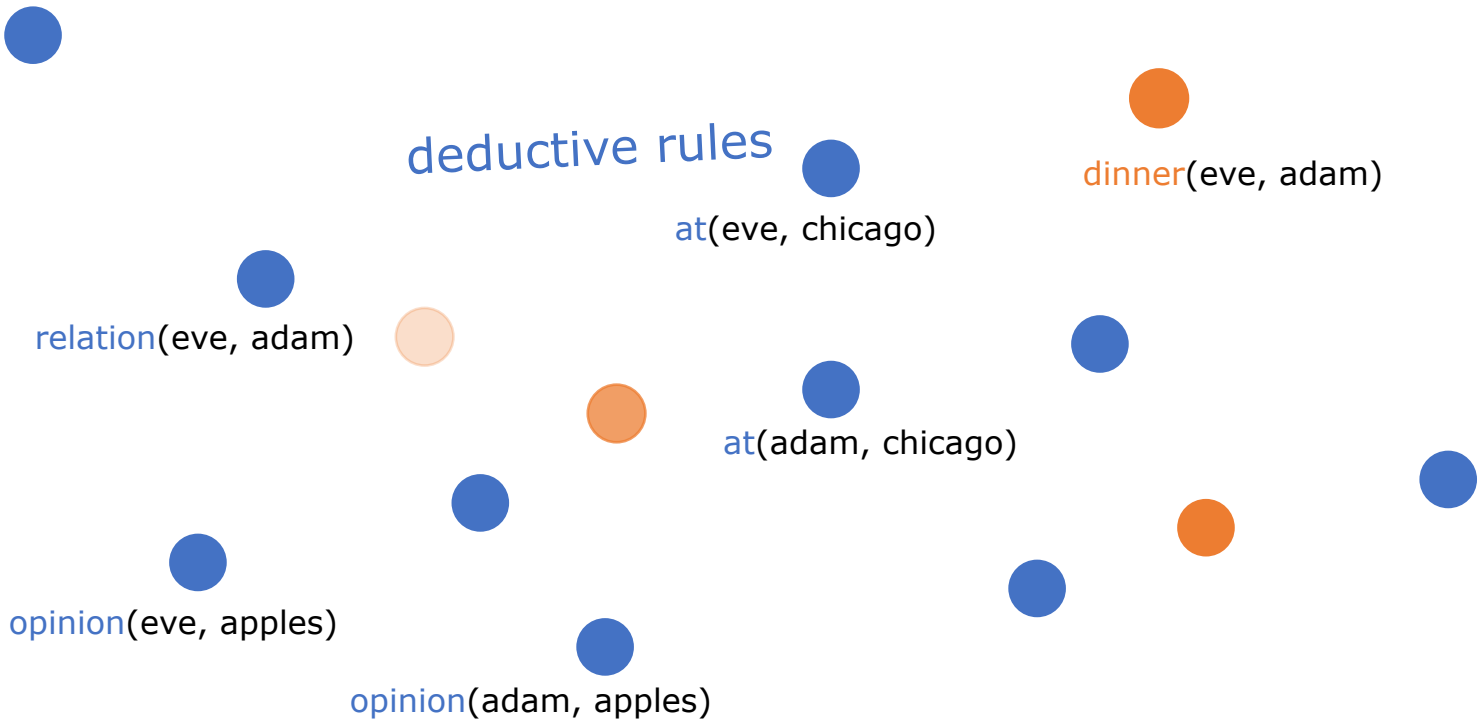


Model How a Database Changes Over Time

200,000 facts right now

50,000 possible events right now

little language
to specify a generative model
of event sequences

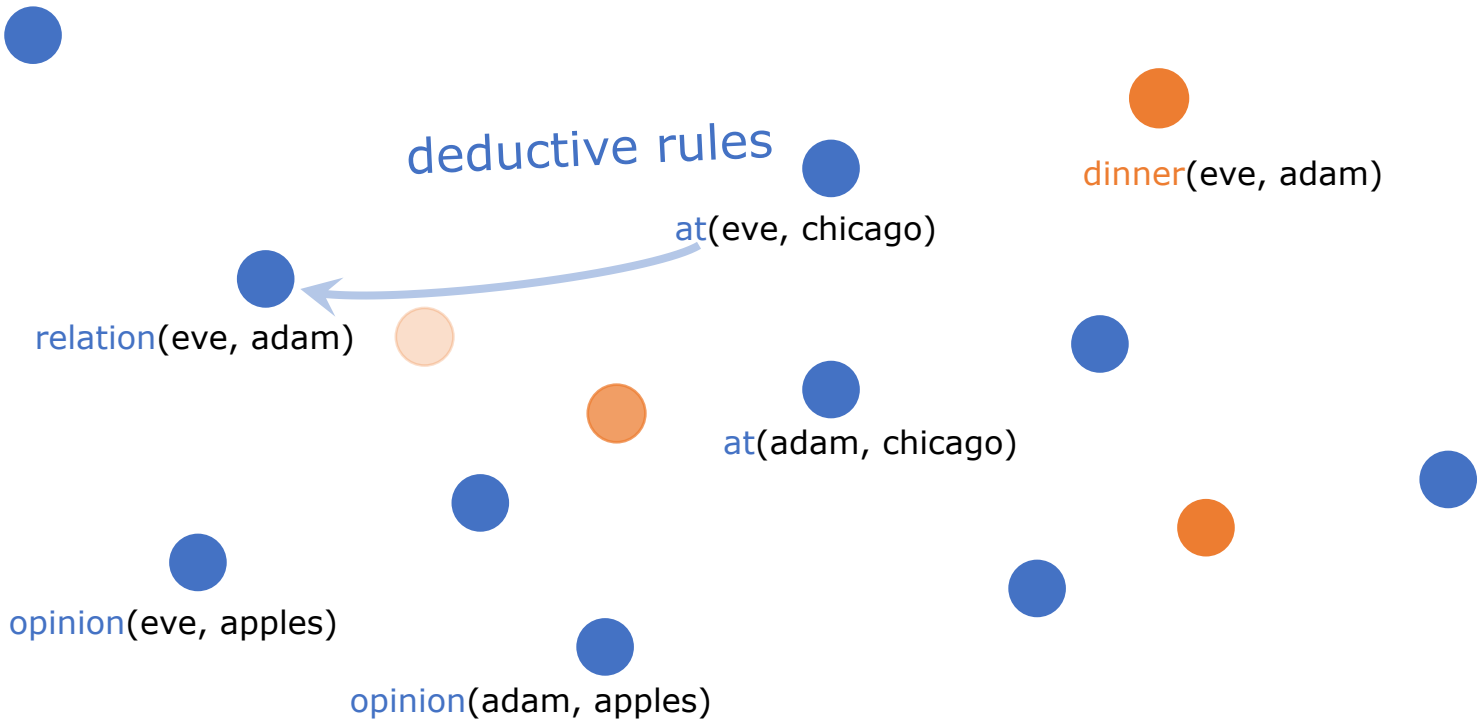


Model How a Database Changes Over Time

200,000 facts right now

50,000 possible events right now

little language
to specify a generative model
of event sequences

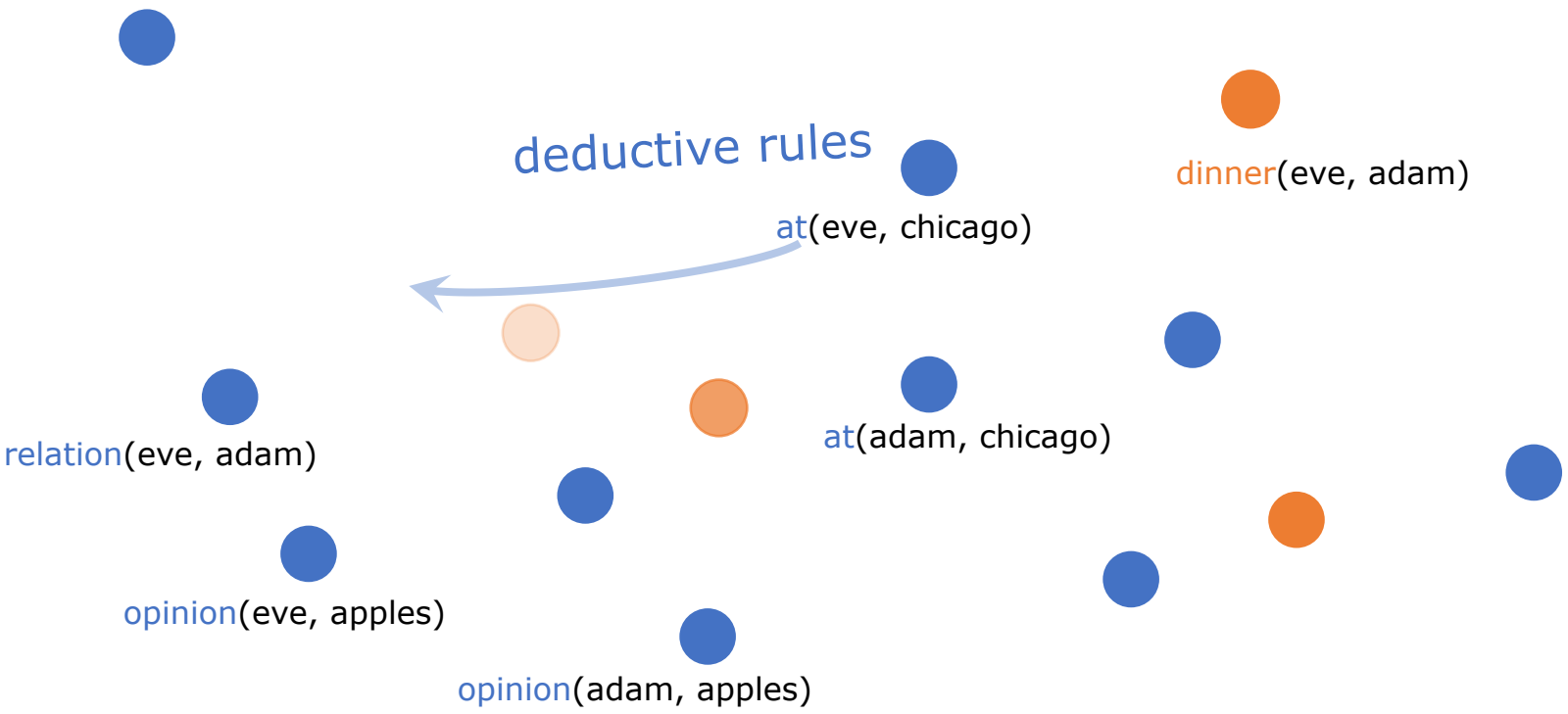


Model How a Database Changes Over Time

200,000 facts right now

50,000 possible events right now

little language
to specify a generative model
of event sequences

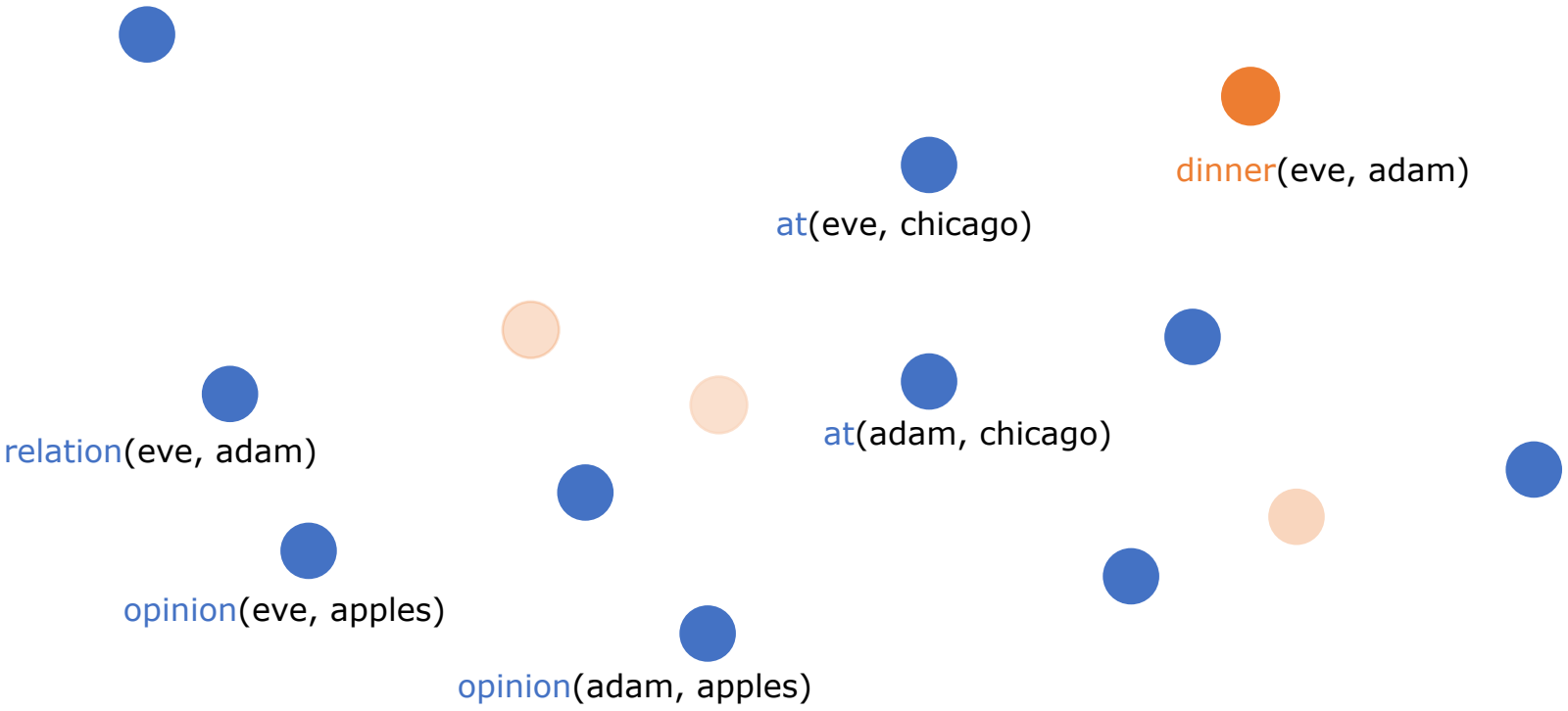


Model How a Database Changes Over Time

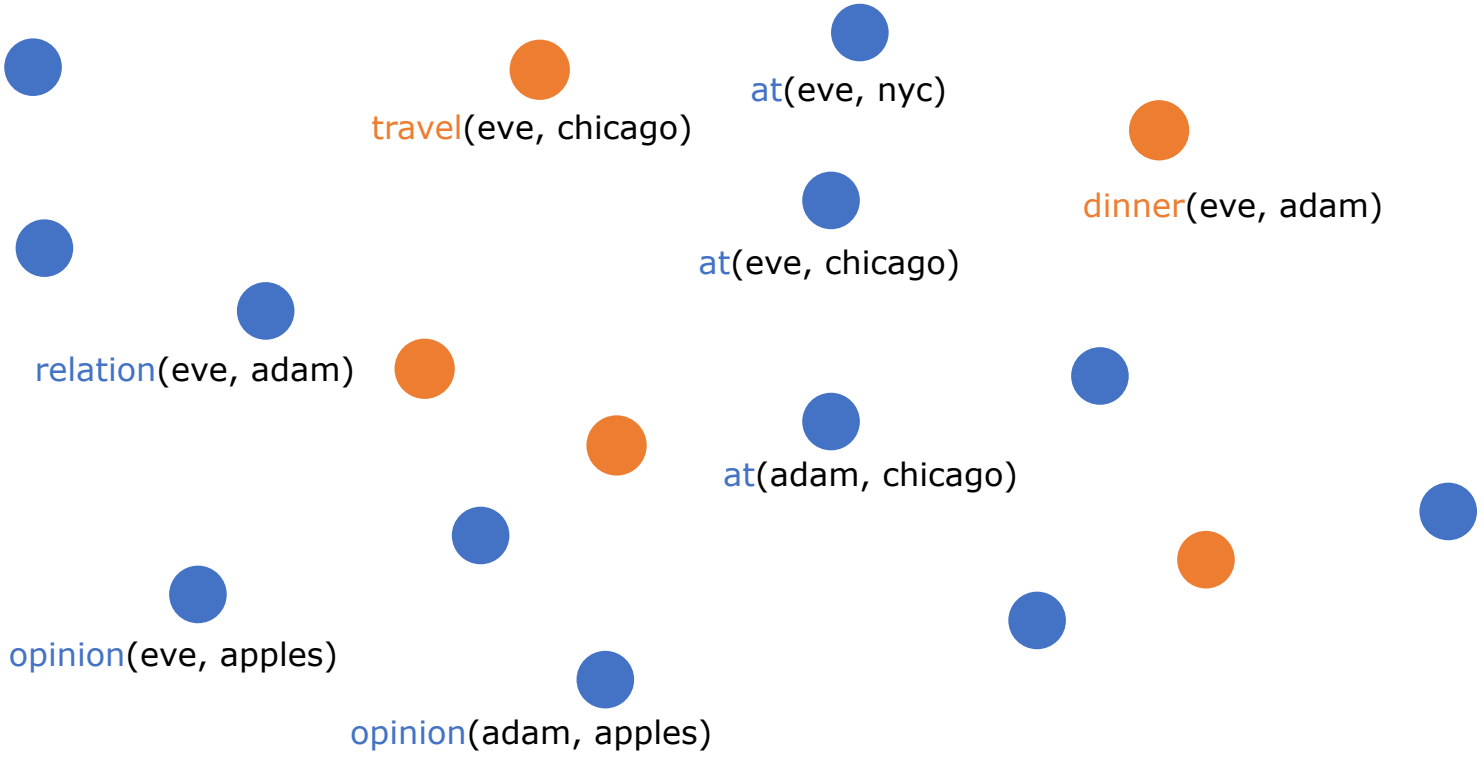
200,000 facts right now

50,000 possible events right now

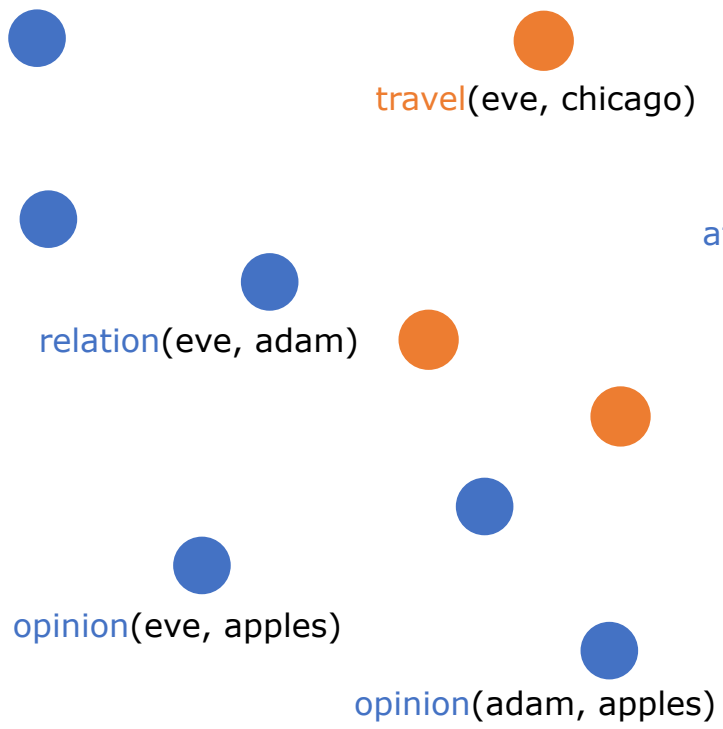
little language
to specify a generative model
of event sequences



Deductive Rules, Triggering Rules



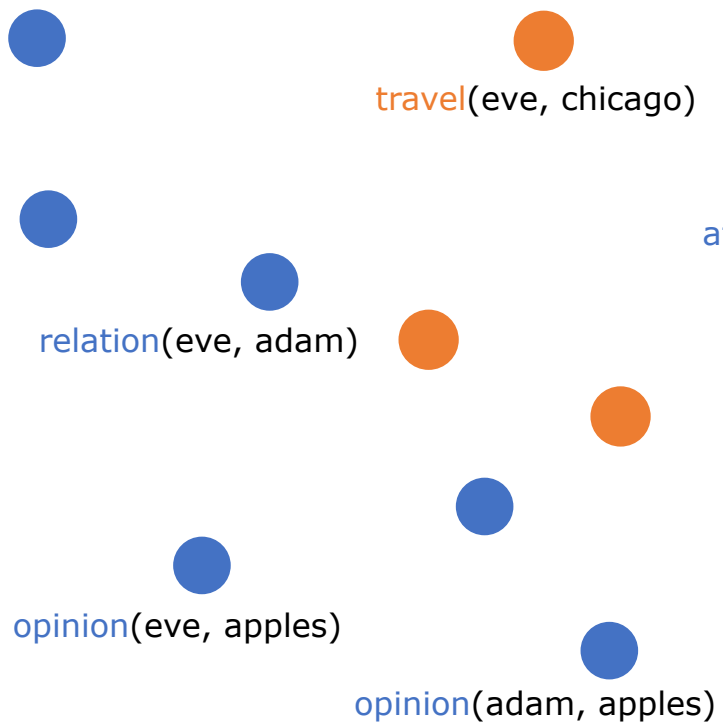
Deductive Rules, Triggering Rules




```
relation(X, Y)
    :- opinion(X, U), opinion(Y, U).

travel(X, P)
    :- relation(X, Y), at(Y, P).

!at(X, Q)
    ← travel(X, P), at(X, Q), P != Q.

at(X, P)
    ← travel(X, P).

dinner(X, Y)
    :- relation(X, Y), at(X, P), at(Y, P).

relation(X, Y)
    ← dinner(X, Y).
```

Deductive Rules, Triggering Rules

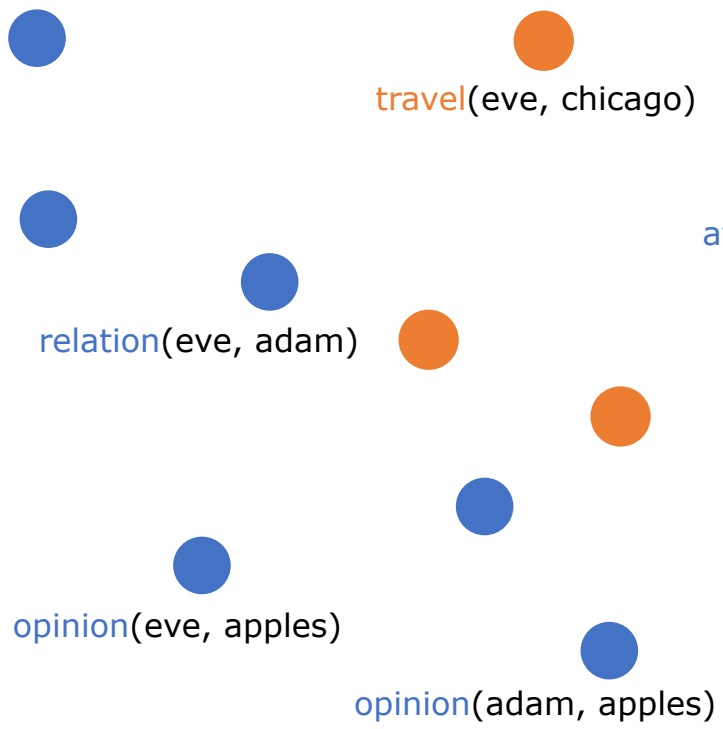
logic!




```
relation(X, Y)
    :- opinion(X, U), opinion(Y, U).

travel(X, P)
    :- relation(X, Y), at(Y, P).

!at(X, Q)
    ← travel(X, P), at(X, Q), P != Q.

at(X, P)
    ← travel(X, P).

dinner(X, Y)
    :- relation(X, Y), at(X, P), at(Y, P).

relation(X, Y)
    ← dinner(X, Y).
```

Deductive Rules, Triggering Rules

which facts are in the database

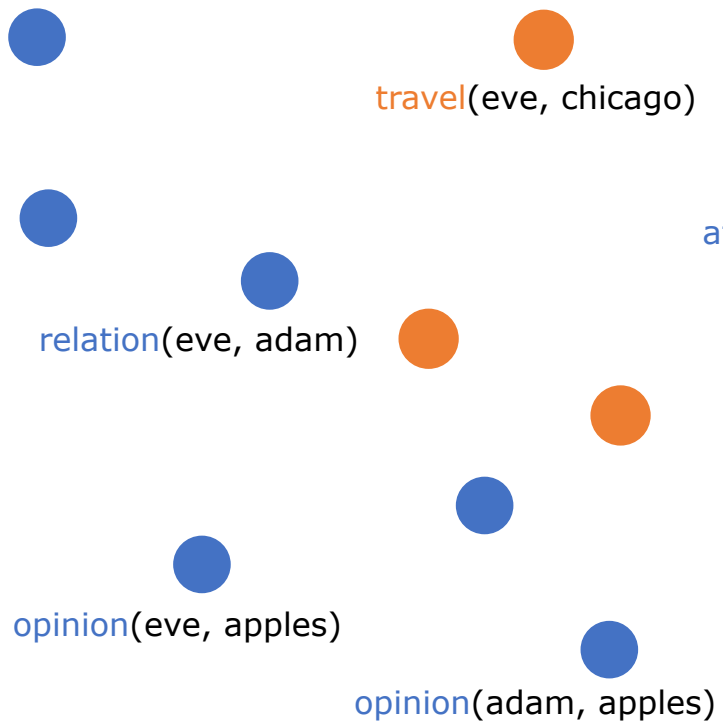


logic!

```
relation(X, Y)
    :- opinion(X, U), opinion(Y, U).

travel(X, P)
    :- relation(X, Y), at(Y, P).

!at(X, Q)
    ← travel(X, P), at(X, Q), P != Q.

at(X, P)
    ← travel(X, P).

dinner(X, Y)
    :- relation(X, Y), at(X, P), at(Y, P).

relation(X, Y)
    ← dinner(X, Y).
```

Deductive Rules, Triggering Rules

which facts are in the database
define a trainable neural architecture



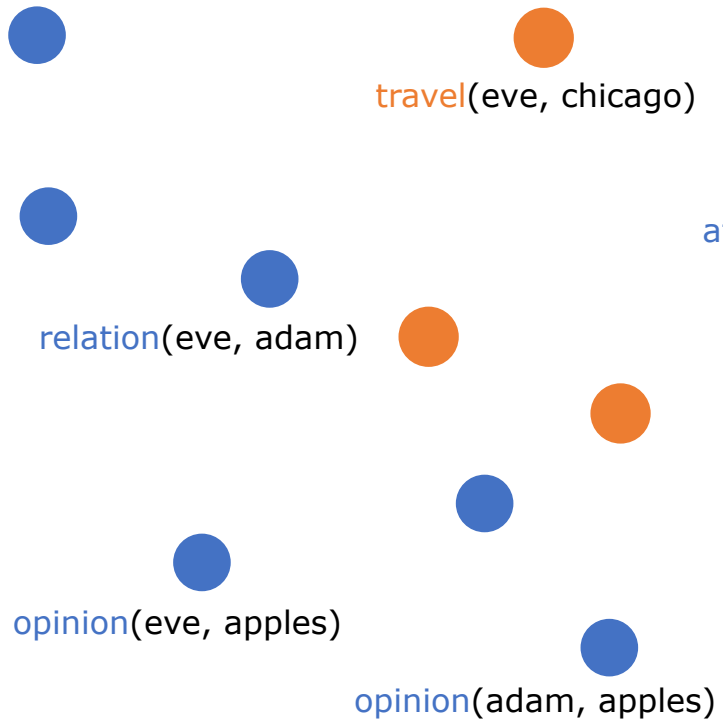
logic!

```
relation(X, Y)
    :- opinion(X, U), opinion(Y, U).

travel(X, P)
    :- relation(X, Y), at(Y, P).

!at(X, Q)
    ← travel(X, P), at(X, Q), P != Q.

at(X, P)
    ← travel(X, P).

dinner(X, Y)
    :- relation(X, Y), at(X, P), at(Y, P).

relation(X, Y)
    ← dinner(X, Y).
```

Deductive Rules, Triggering Rules

which facts are in the database
define a trainable neural architecture
that computes embeddings of the facts

logic!

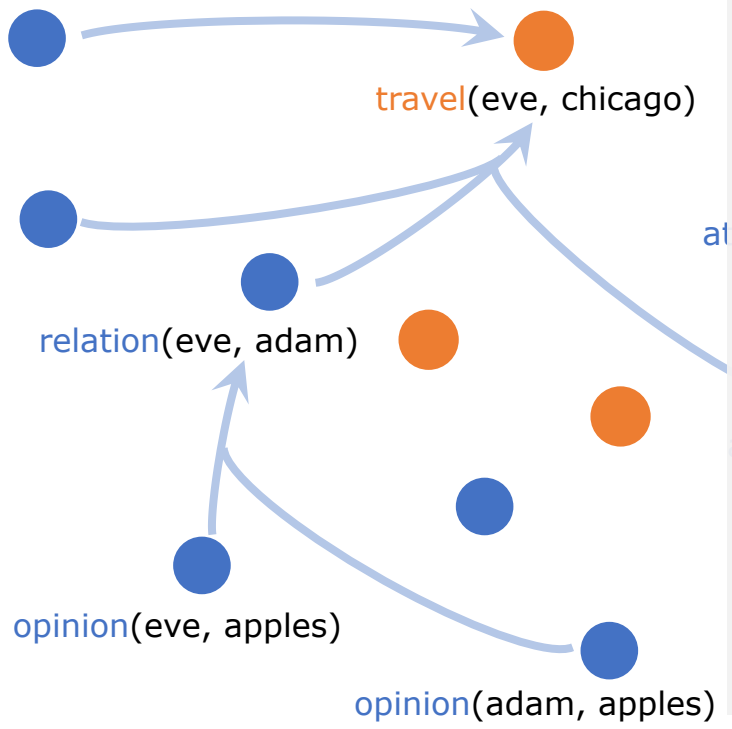



```
relation(X, Y)
    :- opinion(X, U), opinion(Y, U).

travel(X, P)
    :- relation(X, Y), at(Y, P).

!at(X, Q)
    ← travel(X, P), at(X, Q), P != Q.

at(X, P)
    ← travel(X, P).

dinner(X, Y)
    :- relation(X, Y), at(X, P), at(Y, P).

relation(X, Y)
    ← dinner(X, Y).
```

Deductive Rules, Triggering Rules

which facts are in the database
define a trainable neural architecture
that computes embeddings of the facts

logic!

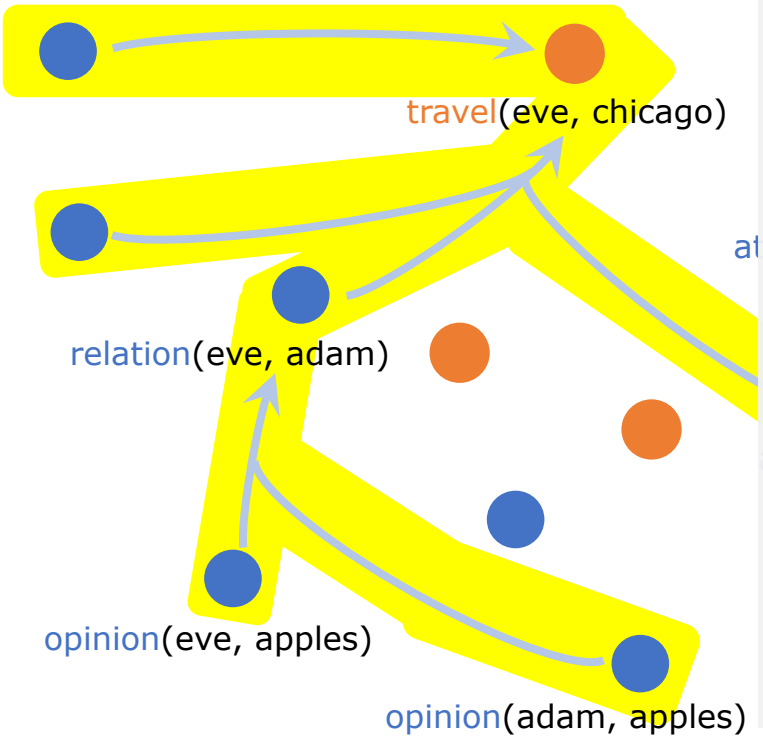



```
relation(X, Y)
    :- opinion(X, U), opinion(Y, U).

travel(X, P)
    :- relation(X, Y), at(Y, P).

!at(X, Q)
    ← travel(X, P), at(X, Q), P != Q.

at(X, P)
    ← travel(X, P).

dinner(X, Y)
    :- relation(X, Y), at(X, P), at(Y, P).

relation(X, Y)
    ← dinner(X, Y).
```

Deductive Rules, Triggering Rules

which facts are in the database
define a trainable neural architecture
that computes embeddings of the facts

logic!

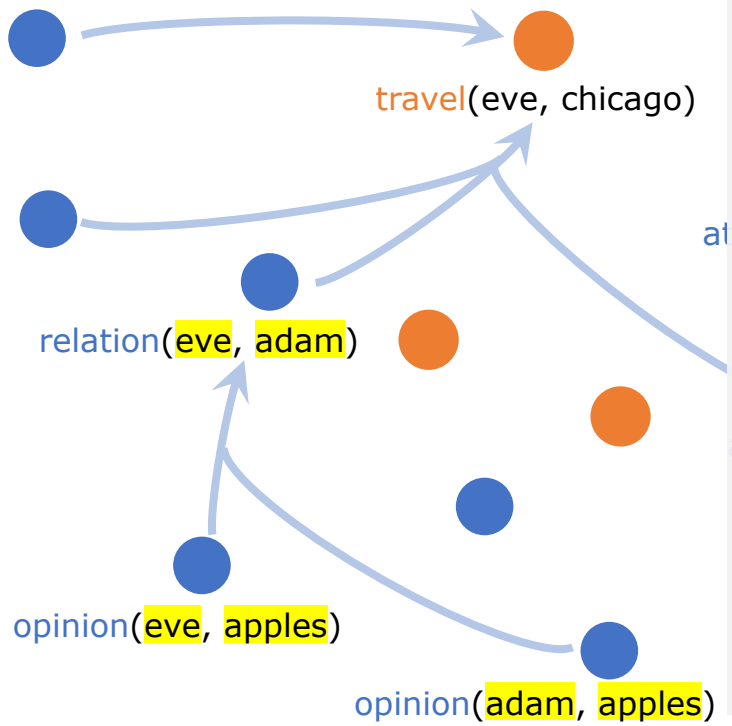



```
relation(X, Y)
    :- opinion(X, U), opinion(Y, U).

travel(X, P)
    :- relation(X, Y), at(Y, P).

!at(X, Q)
    ← travel(X, P), at(X, Q), P != Q.

at(X, P)
    ← travel(X, P).

dinner(X, Y)
    :- relation(X, Y), at(X, P), at(Y, P).

relation(X, Y)
    ← dinner(X, Y).
```

Deductive Rules, Triggering Rules

which facts are in the database
define a trainable neural architecture
that computes embeddings of the facts

logic!




```
relation(X, Y)
    :- opinion(X, U), opinion(Y, U).

travel(X, P)
    :- relation(X, Y), at(Y, P).

!at(X, Q)
    ← travel(X, P), at(X, Q), P != Q.

at(X, P)
    ← travel(X, P).

dinner(X, Y)
    :- relation(X, Y), at(X, P), at(Y, P).

relation(X, Y)
    ← dinner(X, Y).
```

Datalog ➔ Neural Datalog Through Time

Datalog ➔ Neural Datalog Through Time

deductive rule

new fact :- old fact $_1$, old fact $_2$, …

Datalog ➔ Neural Datalog Through Time

deductive rule

add to database

new fact :- old fact $_1$, old fact $_2$, …

Datalog ➔ Neural Datalog Through Time

deductive rule

add to database if

new fact :- old fact $_1$, old fact $_2$, …

Datalog ➔ Neural Datalog Through Time

deductive rule

add to database if these are in database

new fact :- old fact $_1$, old fact $_2$, …

Datalog ➔ Neural Datalog Through Time

deductive rule

add to database if these are in database

new fact :- old fact $_1$, old fact $_2$, …

likes(X, U),

Datalog ➔ Neural Datalog Through Time

deductive rule

add to database if these are in database

new fact :- old fact $_1$, old fact $_2$, …

likes(X, U), likes(Y, U)

Datalog ➔ Neural Datalog Through Time

deductive rule

add to database if these are in database

new fact :- old fact $_1$, old fact $_2$, …

:- likes(X, U), likes(Y, U)

Datalog ➔ Neural Datalog Through Time

deductive rule

add to database if these are in database

new fact :- old fact $_1$, old fact $_2$, …

compatible(X, Y) :- likes(X, U), likes(Y, U)

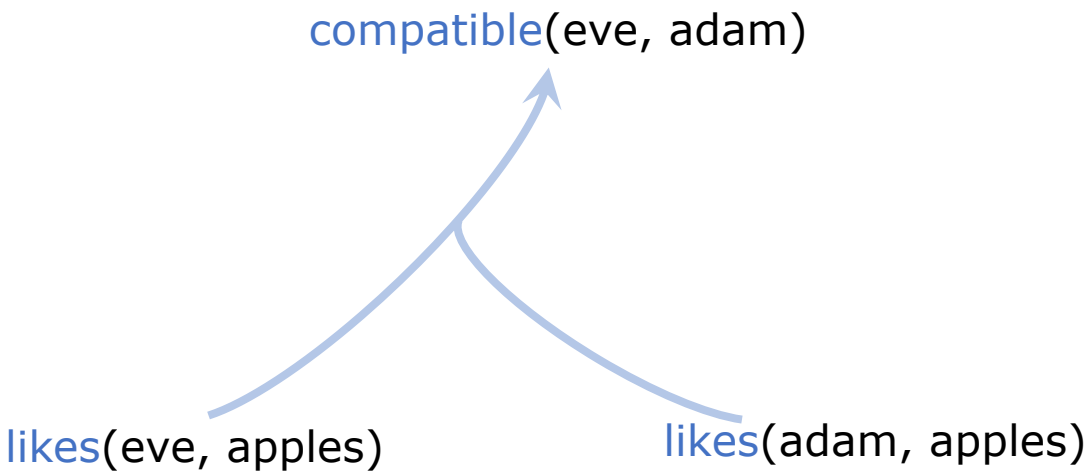
Datalog ➔ Neural Datalog Through Time

deductive rule

add to database if these are in database

new fact :- old fact $_1$, old fact $_2$, …

compatible(X, Y) :- likes(X, U), likes(Y, U)



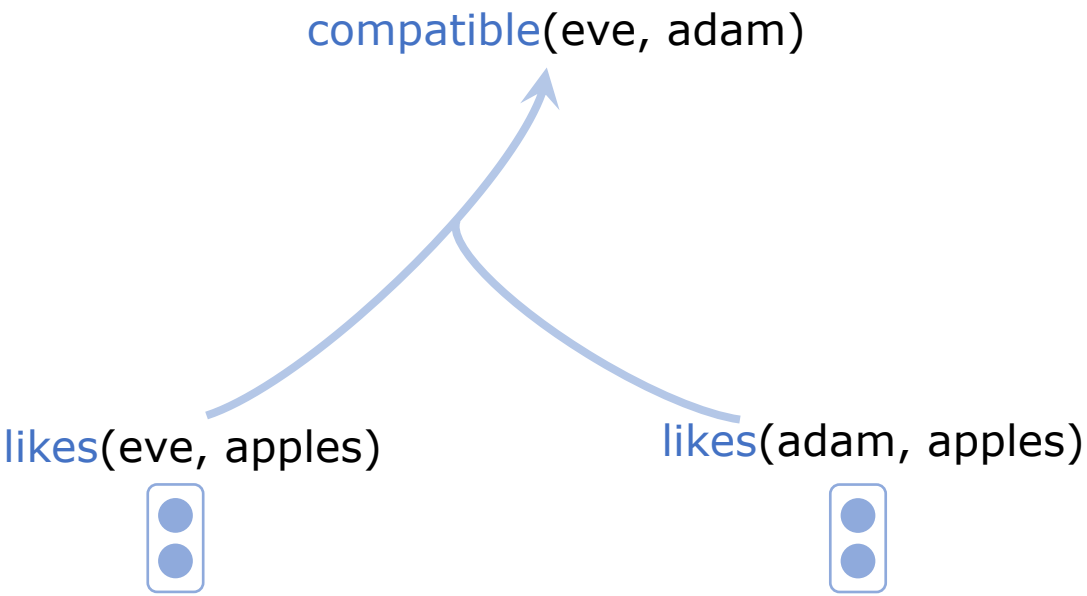
Datalog ➔ Neural Datalog Through Time

deductive rule

add to database if these are in database

new fact :- old fact $_1$, old fact $_2$, …

compatible(X, Y) :- likes(X, U), likes(Y, U)



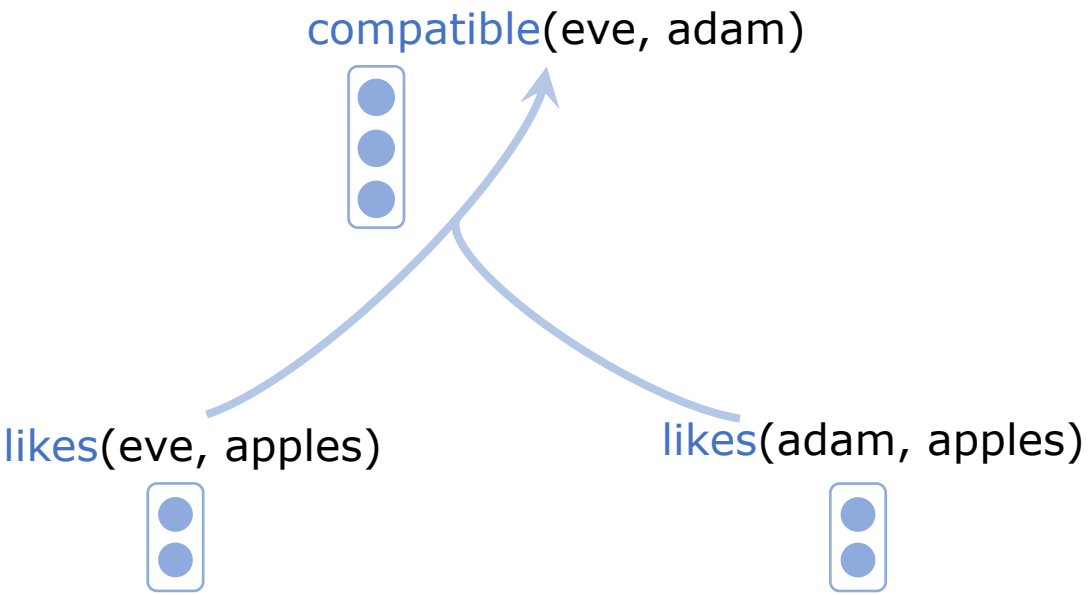
Datalog ➔ Neural Datalog Through Time

deductive rule

add to database if these are in database

new fact :- old fact $_1$, old fact $_2$, …

compatible(X, Y) :- likes(X, U), likes(Y, U)



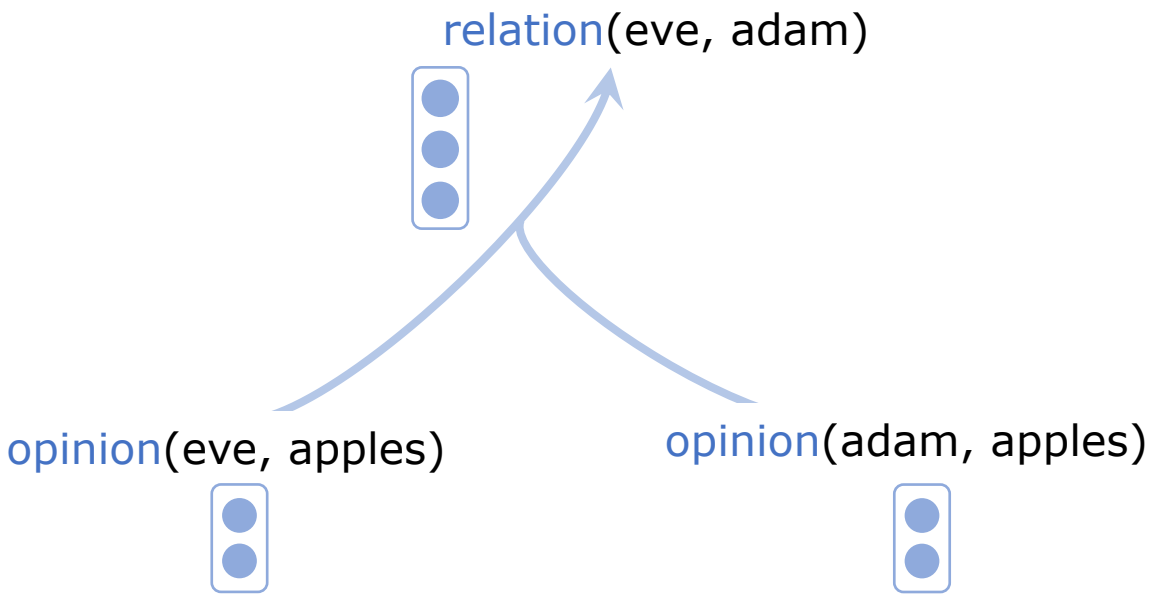
Datalog ➔ Neural Datalog Through Time

deductive rule

add to database if these are in database

new fact :- old fact $_1$, old fact $_2$, …

compatible(X, Y) :- likes(X, U), likes(Y, U)



Datalog ➔ Neural Datalog Through Time

deductive rule

add to database if these are in database

new fact :- old fact $_1$, old fact $_2$, …

triggering rule

new fact ← event, old fact $_1$, old fact $_2$, …

Datalog ➔ Neural Datalog Through Time

deductive rule

add to database if these are in database

new fact :- old fact $_1$, old fact $_2$, …

triggering rule

when

new fact ← event, old fact $_1$, old fact $_2$, …

Datalog → Neural Datalog Through Time

deductive rule

add to database if these are in database

new fact :- old fact $_1$, old fact $_2$, …

triggering rule

when this happens

new fact ← event, old fact $_1$, old fact $_2$, …

Datalog ➔ Neural Datalog Through Time

deductive rule

add to database if these are in database

new fact :- old fact $_1$, old fact $_2$, …

triggering rule

when this happens while these are in database

new fact ← event, old fact $_1$, old fact $_2$, …

Datalog ➔ Neural Datalog Through Time

deductive rule

add to database if these are in database

new fact :- old fact $_1$, old fact $_2$, …

triggering rule

add to database when this happens while these are in database

new fact ← event, old fact $_1$, old fact $_2$, …

Datalog → Neural Datalog Through Time

deductive rule

add to database if these are in database

new fact :- old fact $_1$, old fact $_2$, …

triggering rule

add to database when this happens while these are in database

new fact ← event, old fact $_1$, old fact $_2$, …

! old fact ← event, old fact $_1$, old fact $_2$, …

Datalog ➔ Neural Datalog Through Time

deductive rule

add to database if these are in database

new fact :- old fact $_1$, old fact $_2$, …

triggering rule

add to database when this happens while these are in database

new fact ← event, old fact $_1$, old fact $_2$, …

when

! old fact ← event, old fact $_1$, old fact $_2$, …

Datalog ➔ Neural Datalog Through Time

deductive rule

add to database if these are in database

new fact :- old fact $_1$, old fact $_2$, …

triggering rule

add to database when this happens while these are in database

new fact ← event, old fact $_1$, old fact $_2$, …

when this happens

! old fact ← event, old fact $_1$, old fact $_2$, …

Datalog ➔ Neural Datalog Through Time

deductive rule

add to database | if | these are in database

new fact :- old fact $_1$, old fact $_2$, …

triggering rule

add to database | when this happens | while these are in database

new fact ← event, old fact $_1$, old fact $_2$, …

when this happens | while these are in database

! old fact ← event, old fact $_1$, old fact $_2$, …

Datalog ➔ Neural Datalog Through Time

deductive rule

add to database if these are in database

new fact :- old fact $_1$, old fact $_2$, …

triggering rule

add to database when this happens while these are in database

new fact ← event, old fact $_1$, old fact $_2$, …

delete when this happens while these are in database

! old fact ← event, old fact $_1$, old fact $_2$, …

Computing the Embeddings

Computing the Embeddings

relation(eve, adam)

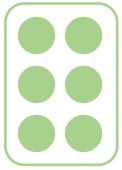
Computing the Embeddings

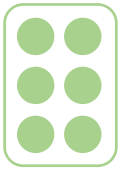
relation(X, Y) :- opinion(X, U), opinion(Y, U)

relation(eve, adam)

Computing the Embeddings

relation(X, Y) :- opinion(X, U), opinion(Y, U)

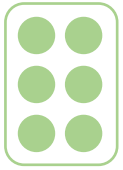


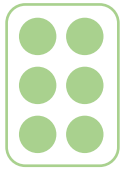


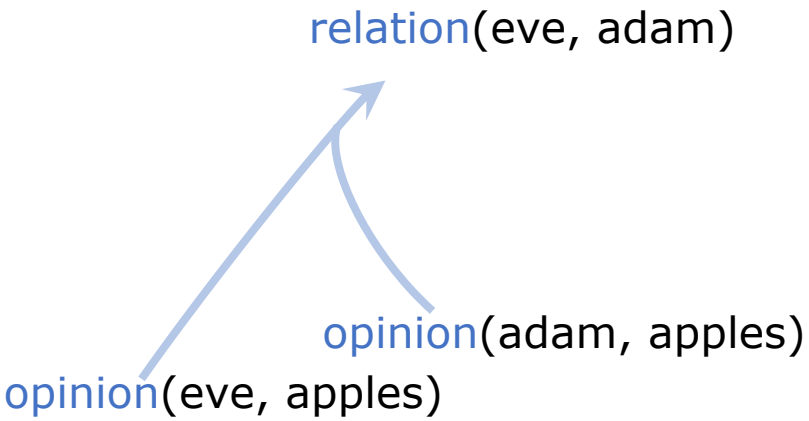
relation(eve, adam)

Computing the Embeddings

relation(X, Y) :- opinion(X, U), opinion(Y, U)

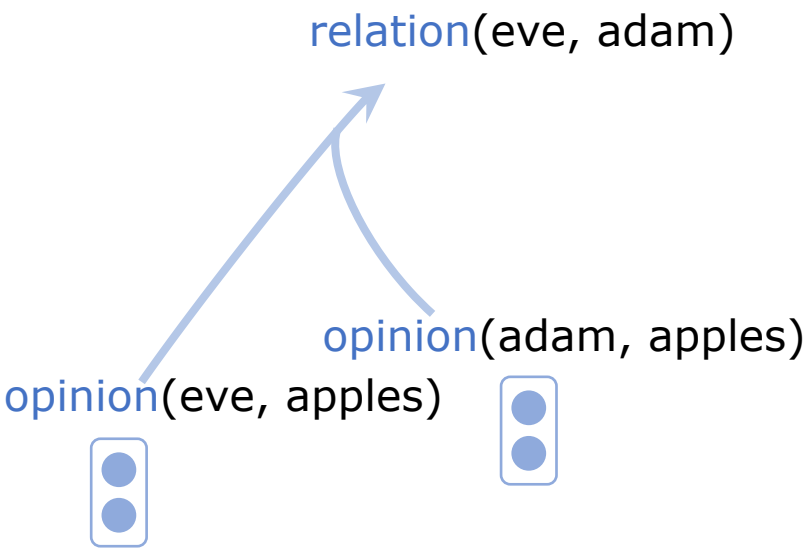






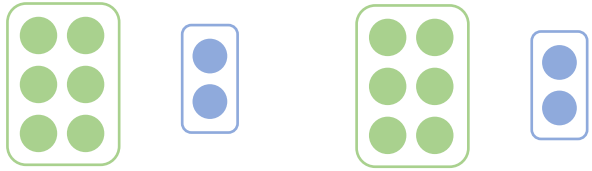
Computing the Embeddings

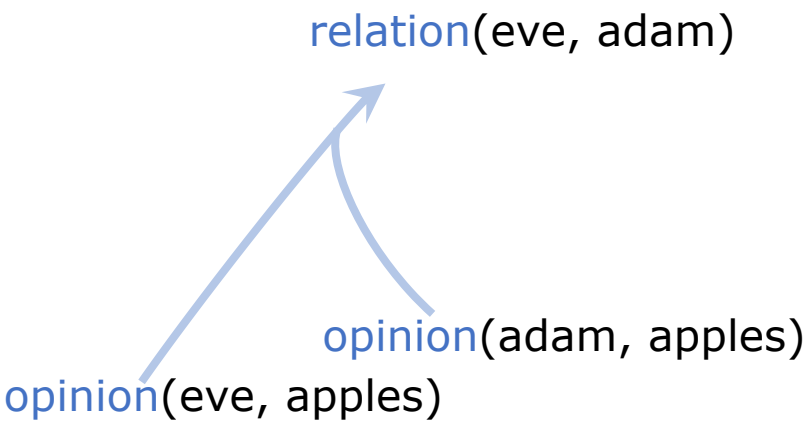
relation(X, Y) :- opinion(X, U), opinion(Y, U)



Computing the Embeddings

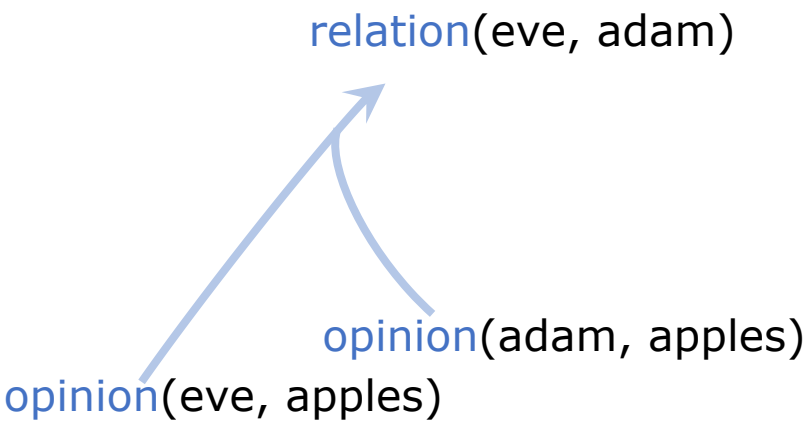
relation(X, Y) :- opinion(X, U), opinion(Y, U)





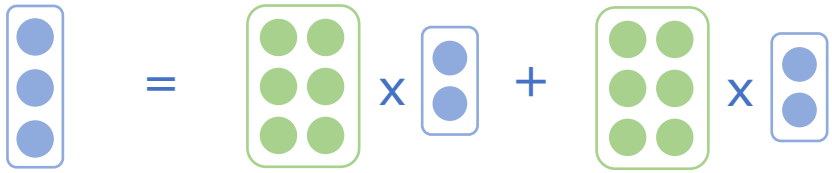
Computing the Embeddings

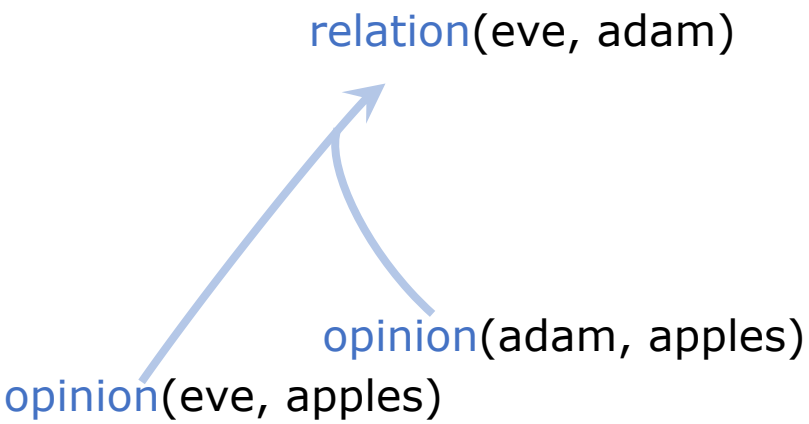
relation(X, Y) :- opinion(X, U), opinion(Y, U)



Computing the Embeddings

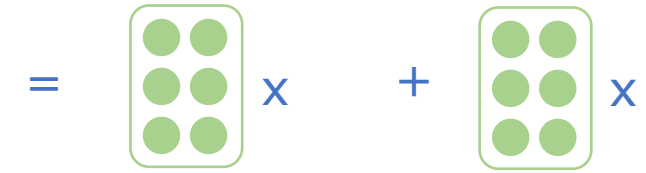
relation(X, Y) :- opinion(X, U), opinion(Y, U)

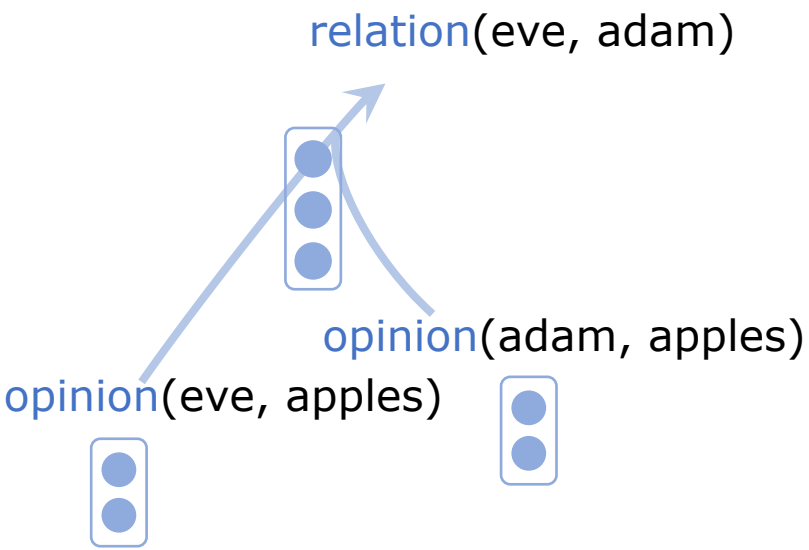




Computing the Embeddings

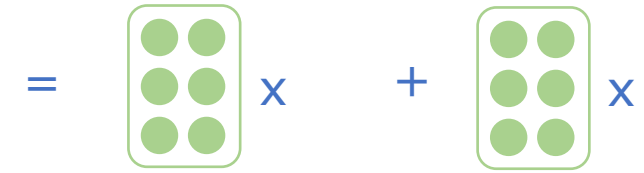
relation(X, Y) :- opinion(X, U), opinion(Y, U)

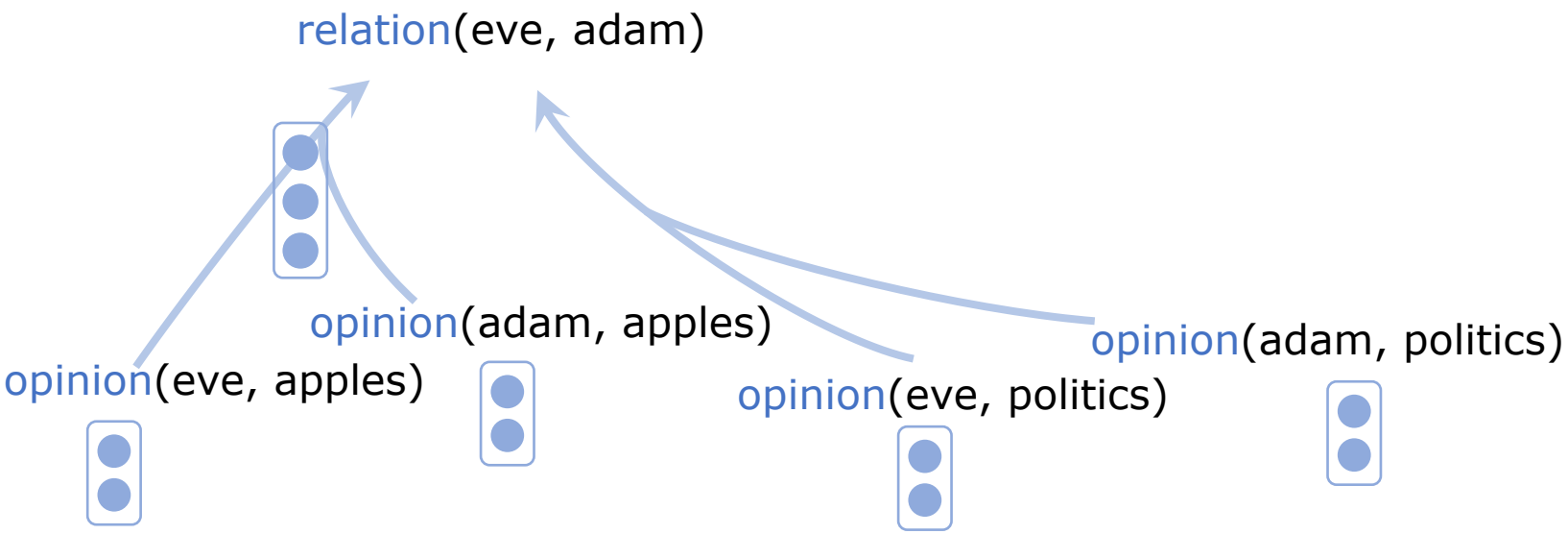




Computing the Embeddings

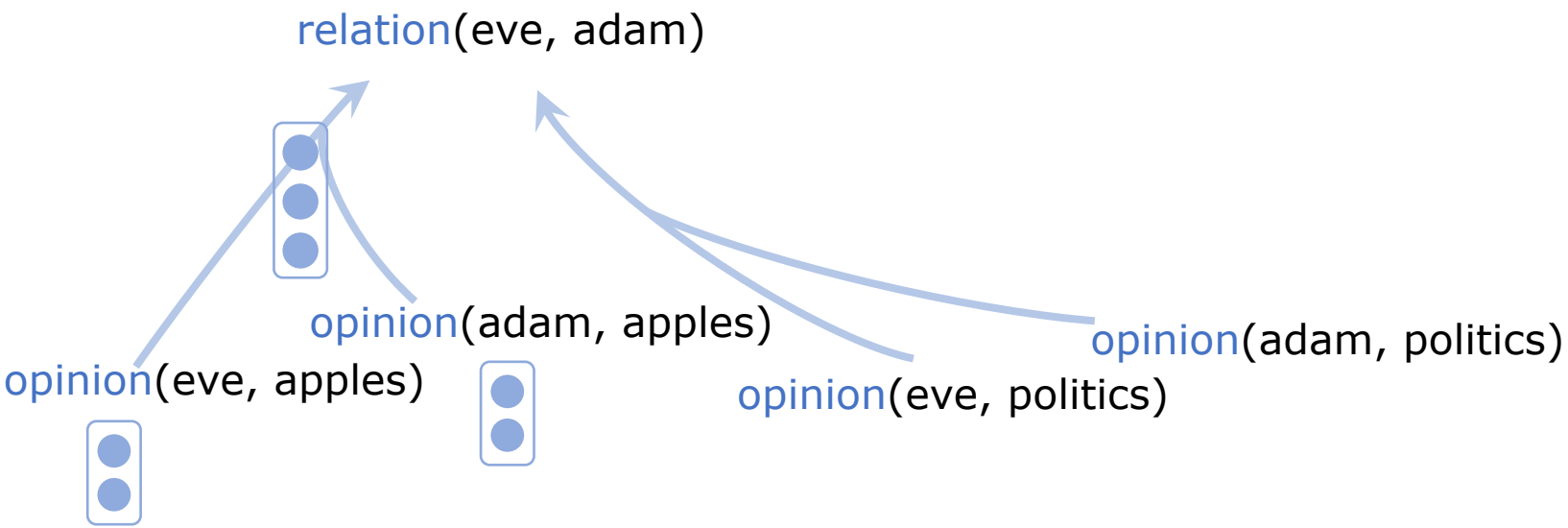
relation(X, Y) :- opinion(X, U), opinion(Y, U)





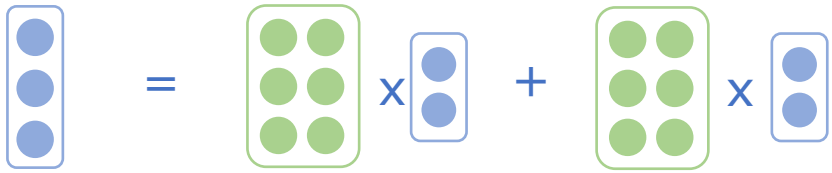
Computing the Embeddings

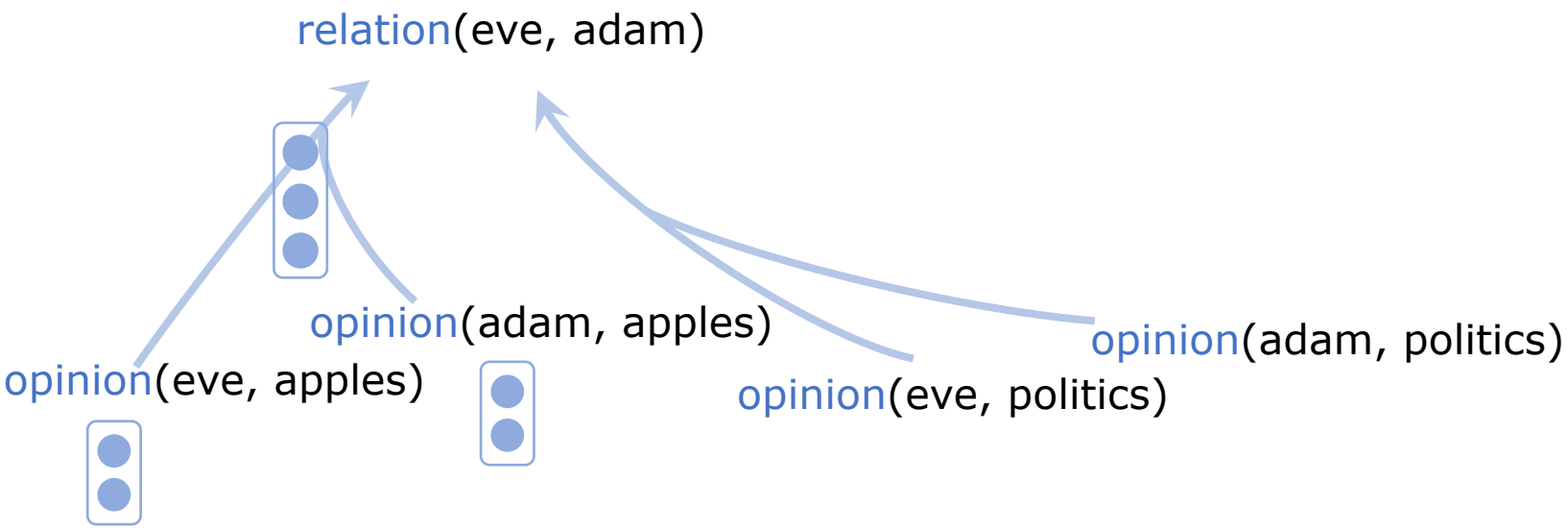
relation(X, Y) :- opinion(X, U), opinion(Y, U)



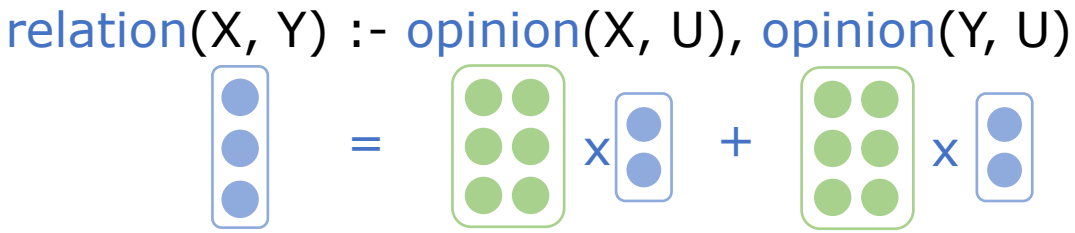
Computing the Embeddings

relation(X, Y) :- opinion(X, U), opinion(Y, U)



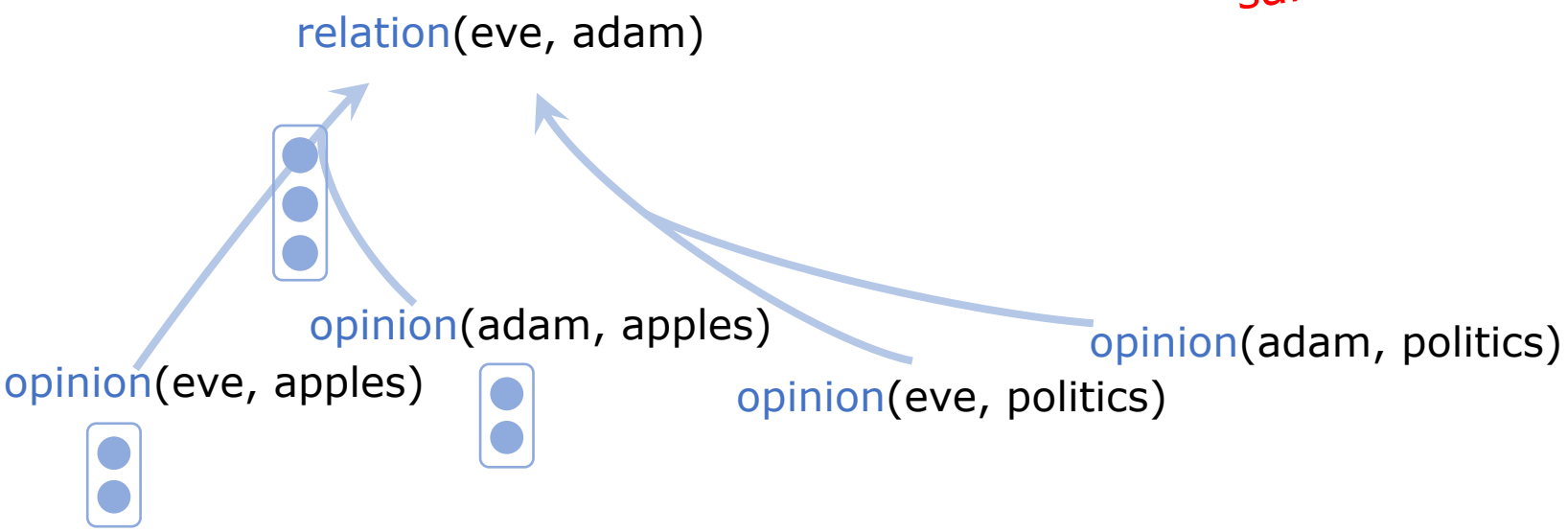


Computing the Embeddings



different inputs

same params

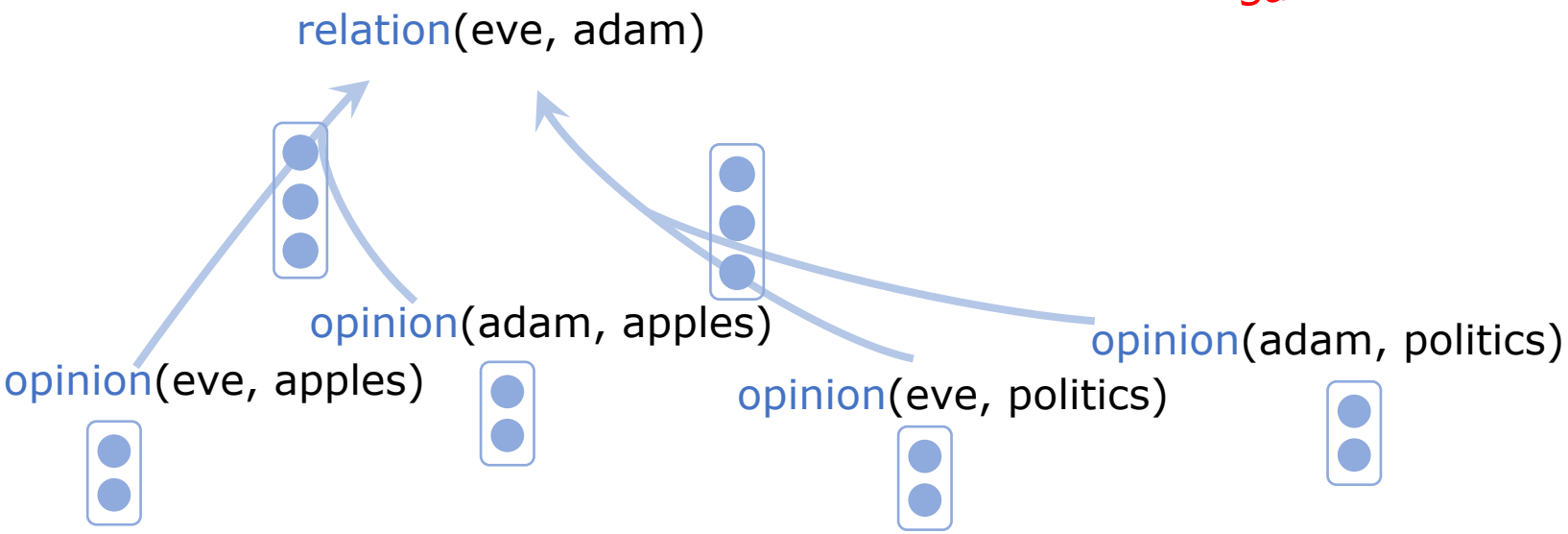


Computing the Embeddings

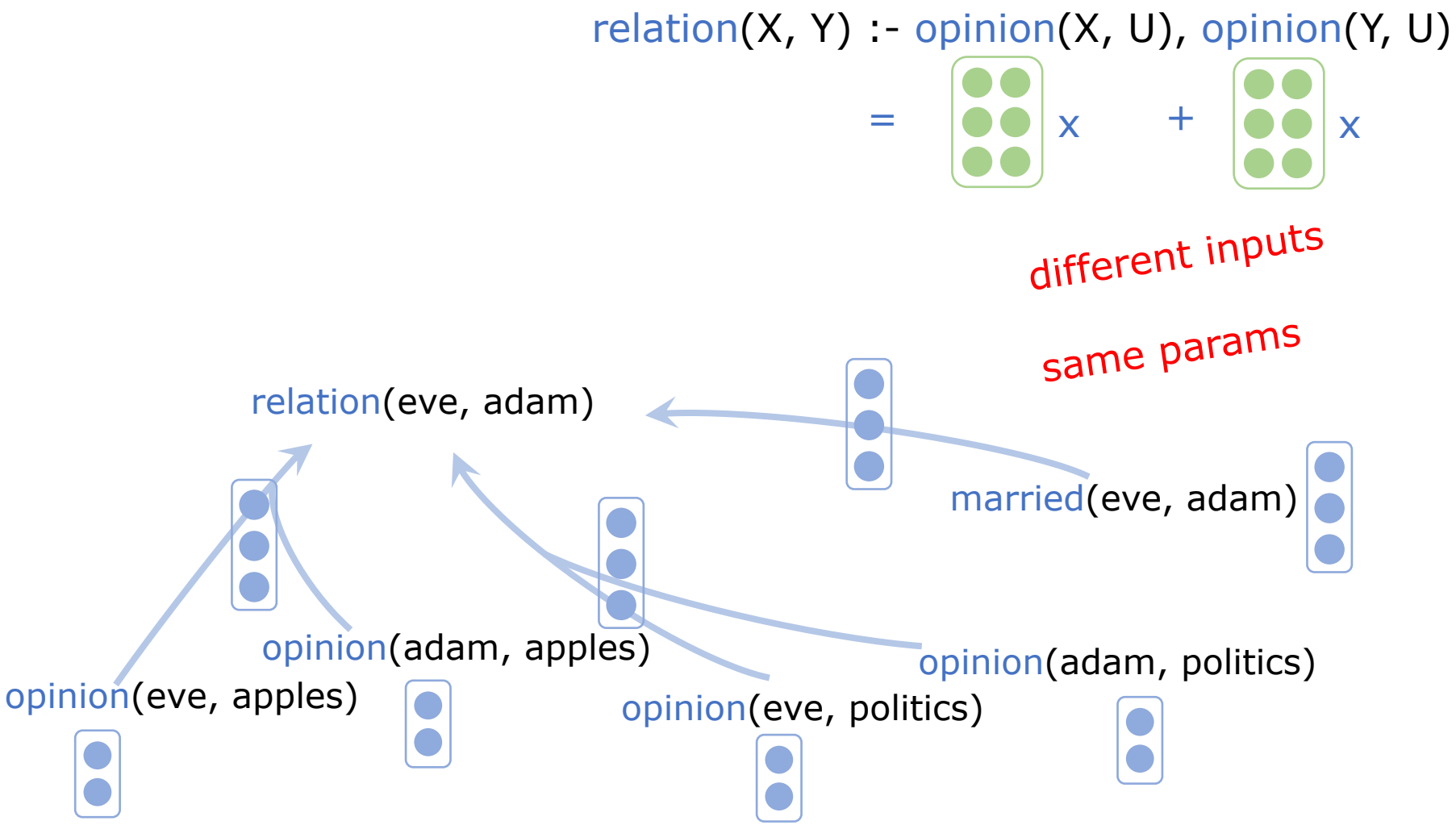
relation(X, Y) :- opinion(X, U), opinion(Y, U)

different inputs

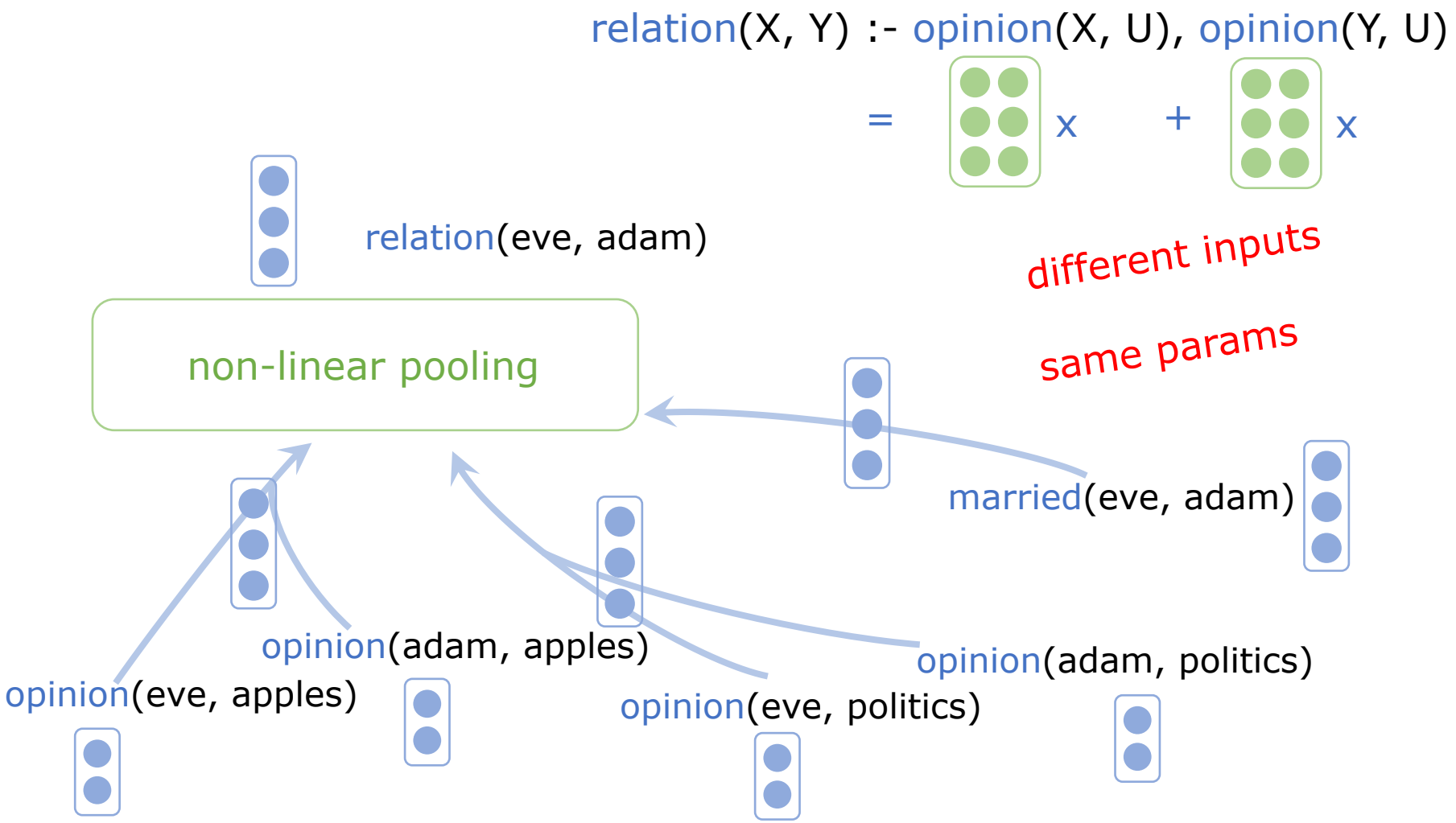
same params



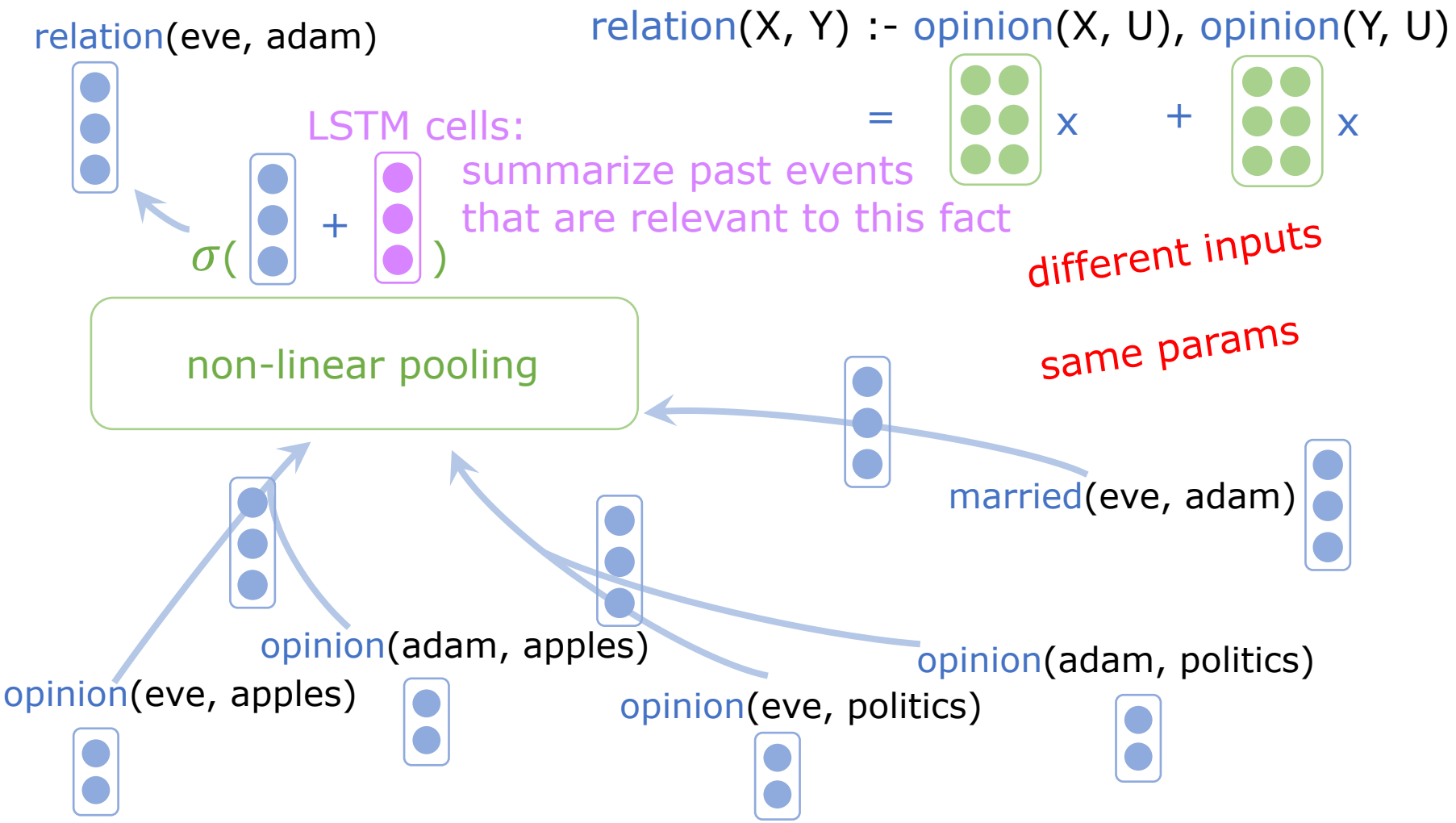
Computing the Embeddings



Computing the Embeddings



Computing the Embeddings



Computing Embeddings & Probabilities

Computing Embeddings & Probabilities

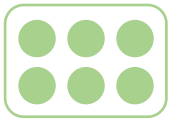
...

relation(eve, adam)

relation(eve, cain)

Computing Embeddings & Probabilities

travel(X, P) :- relation(X, Y), at(Y, P).



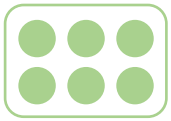
...

relation(eve, adam)

relation(eve, cain)

Computing Embeddings & Probabilities

travel(X, P) :- relation(X, Y), at(Y, P).



...

...

relation(eve, adam)

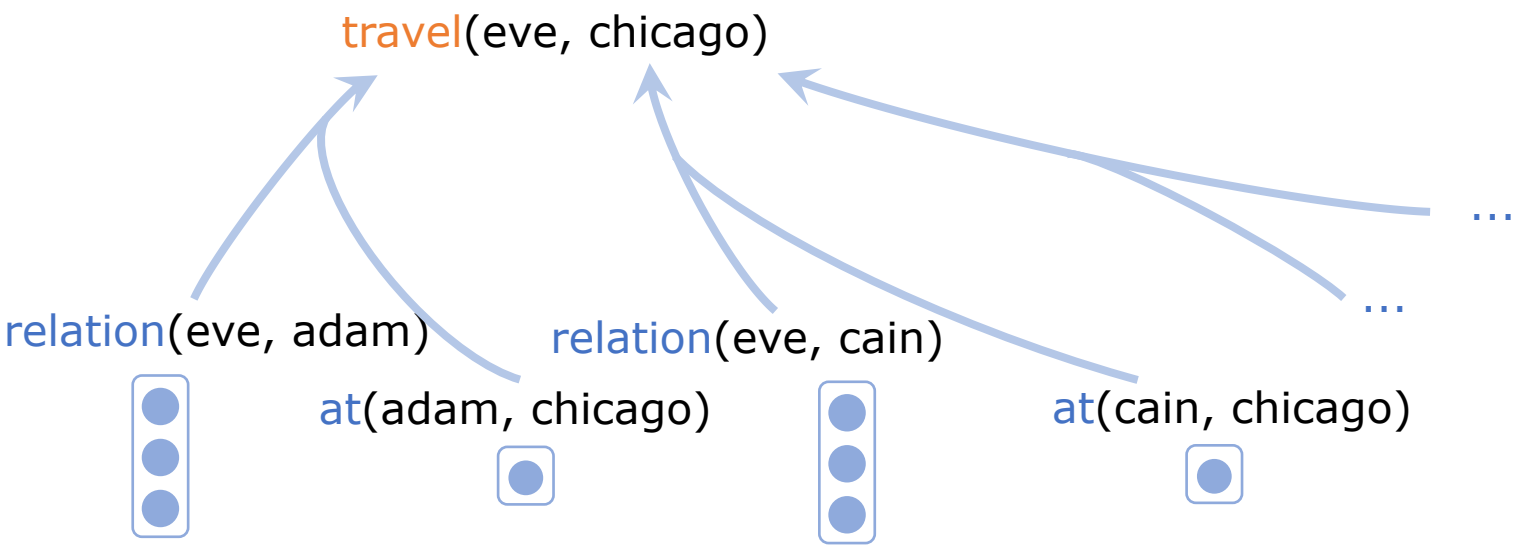
at(adam, chicago)

relation(eve, cain)

at(cain, chicago)

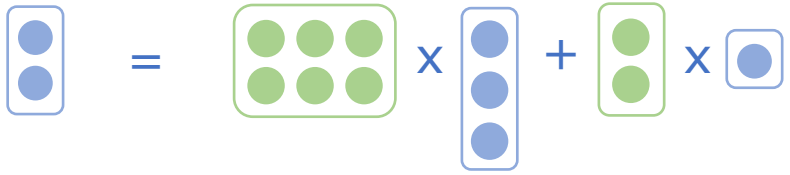
Computing Embeddings & Probabilities

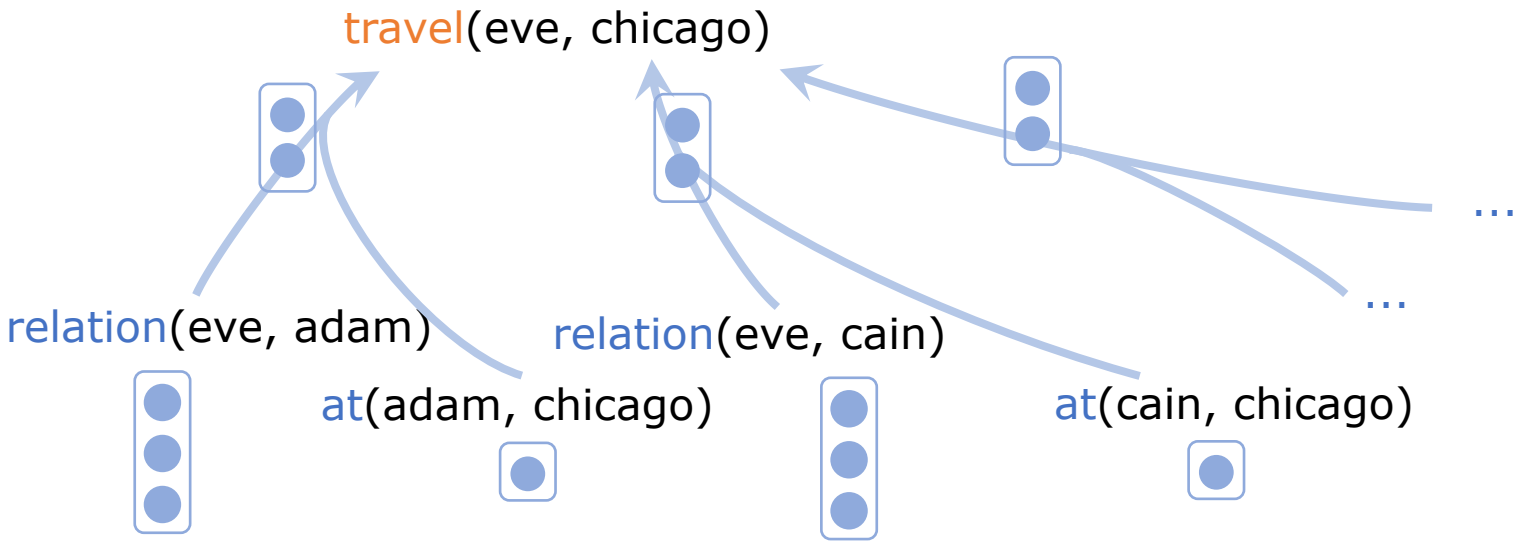
travel(X, P) :- relation(X, Y), at(Y, P).



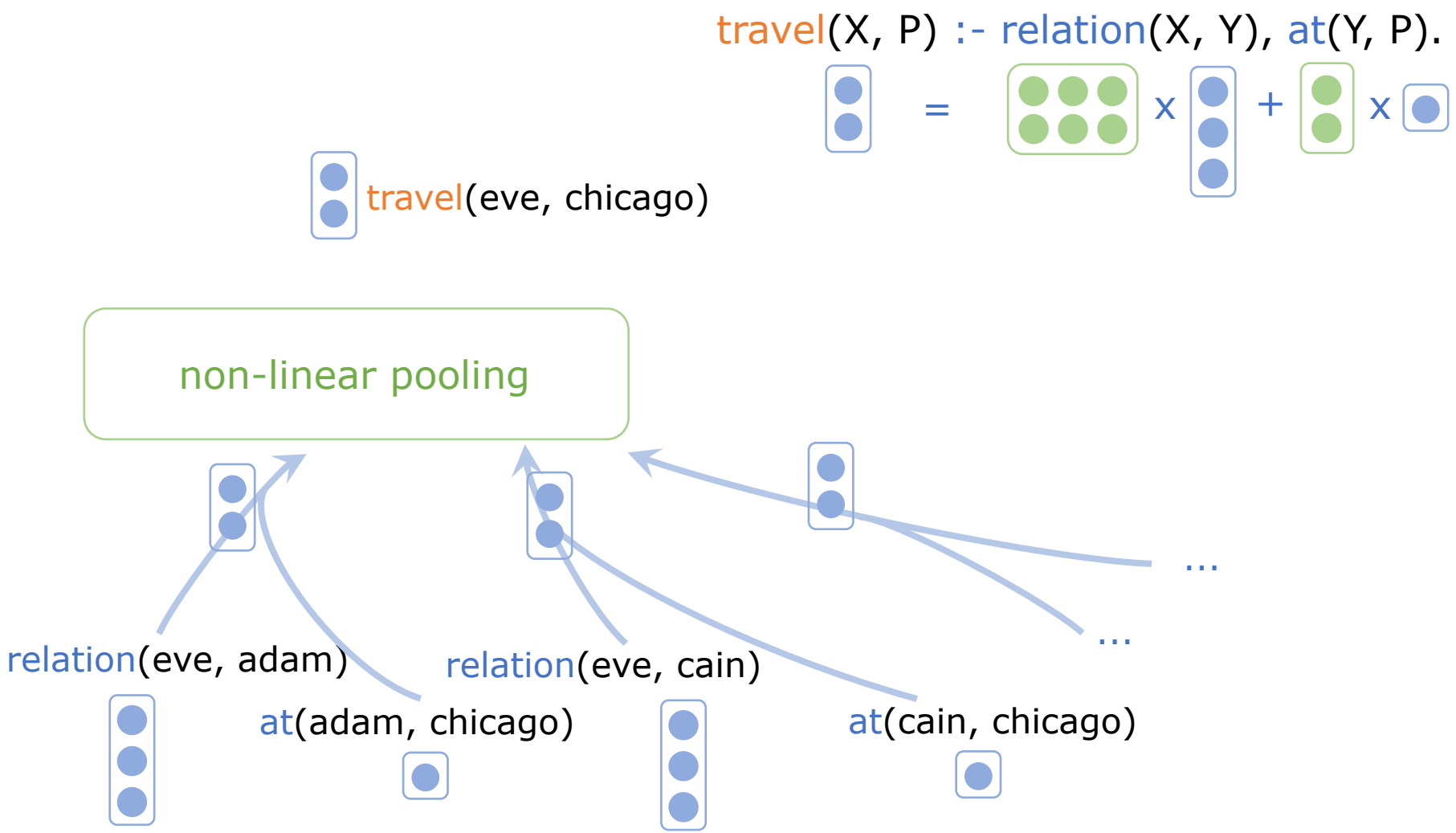
Computing Embeddings & Probabilities

travel(X, P) :- relation(X, Y), at(Y, P).

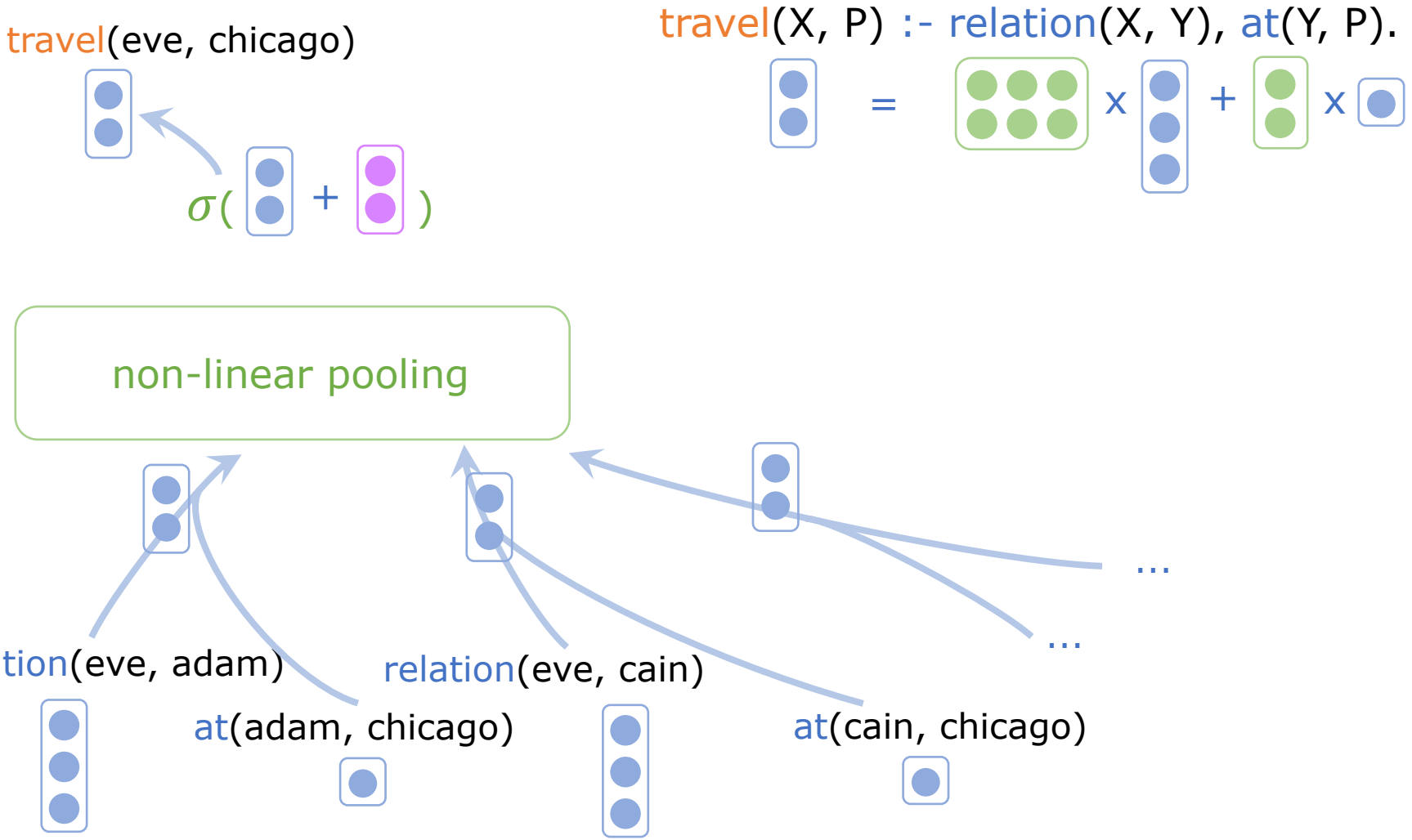




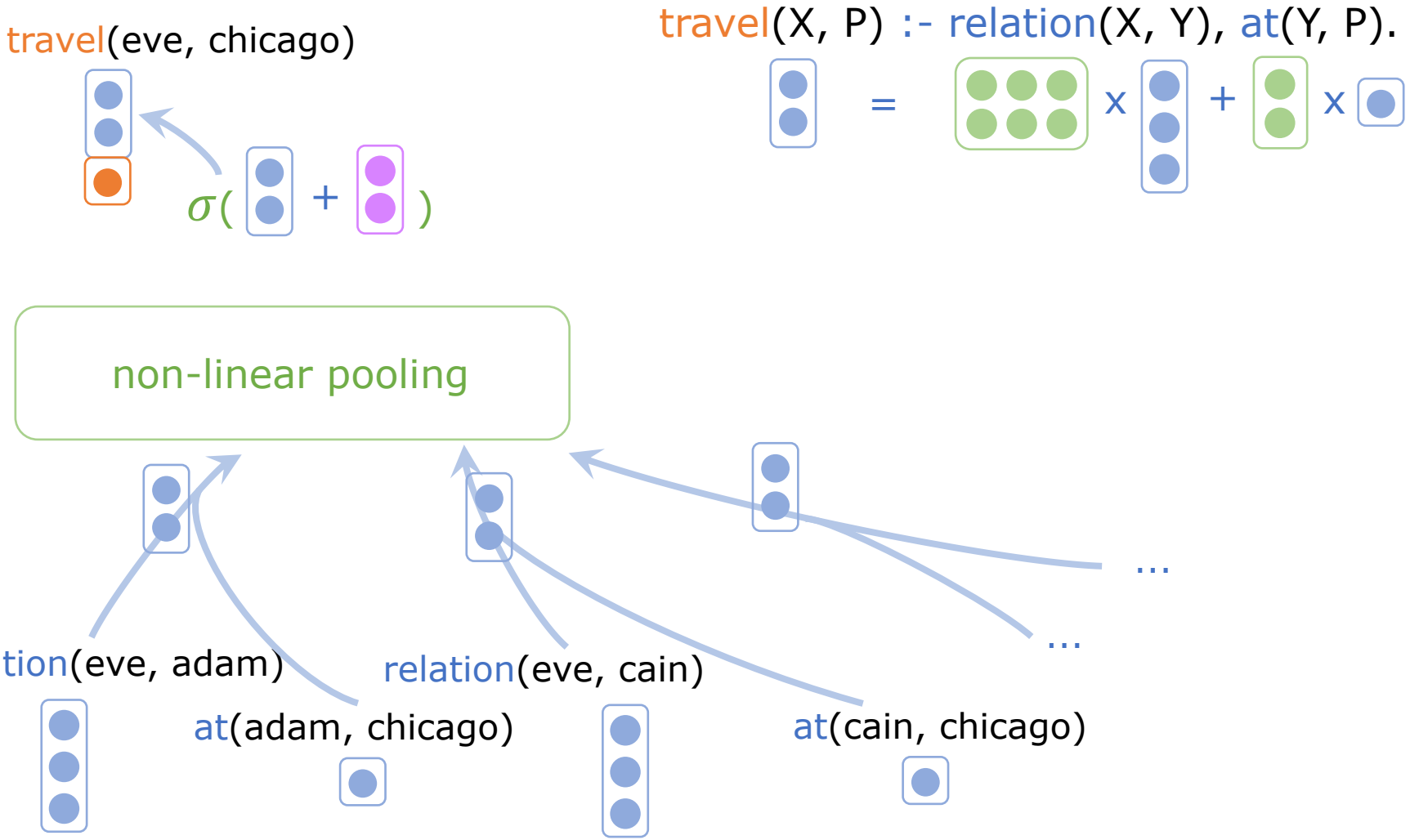
Computing Embeddings & Probabilities



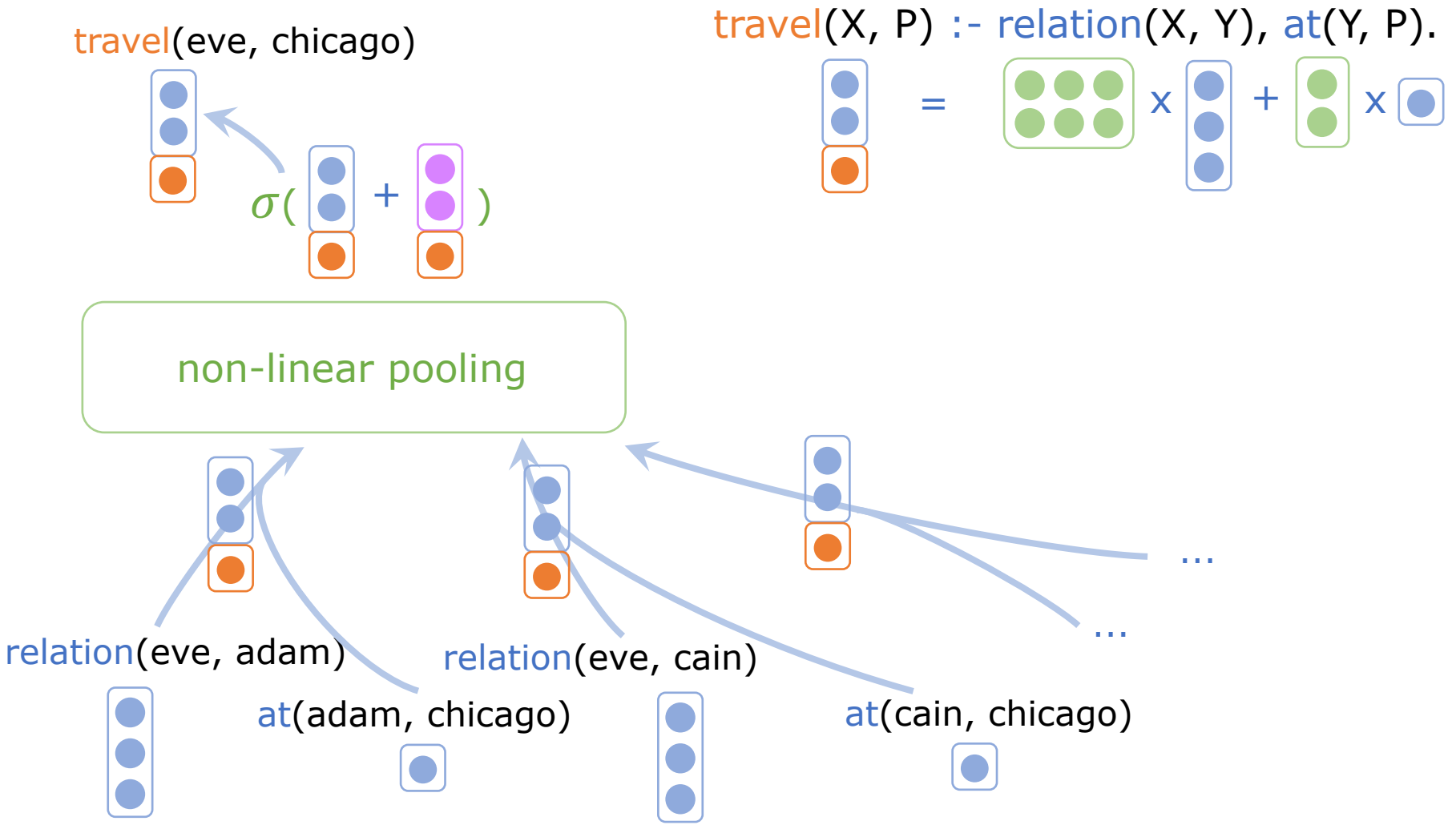
Computing Embeddings & Probabilities



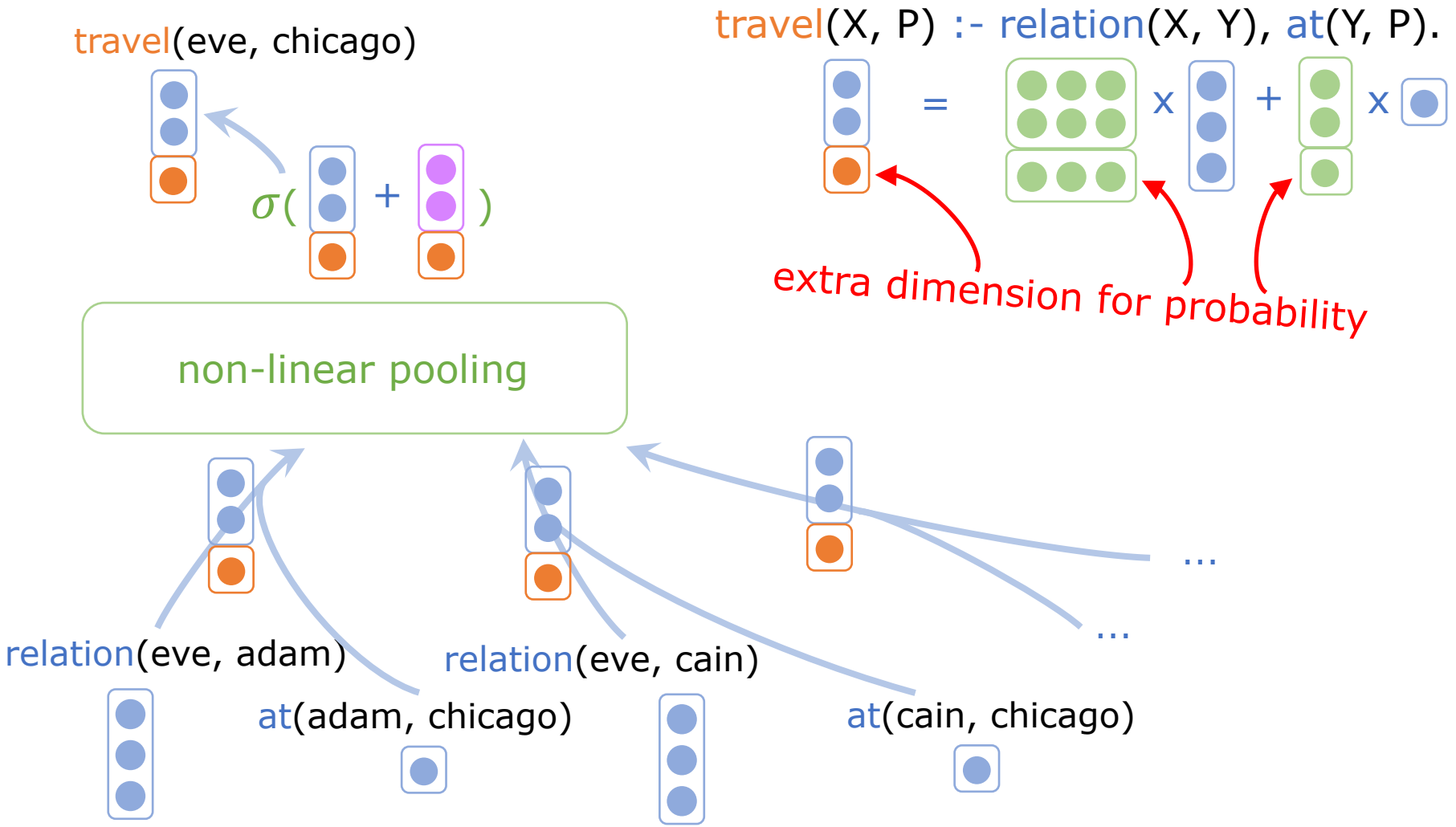
Computing Embeddings & Probabilities



Computing Embeddings & Probabilities

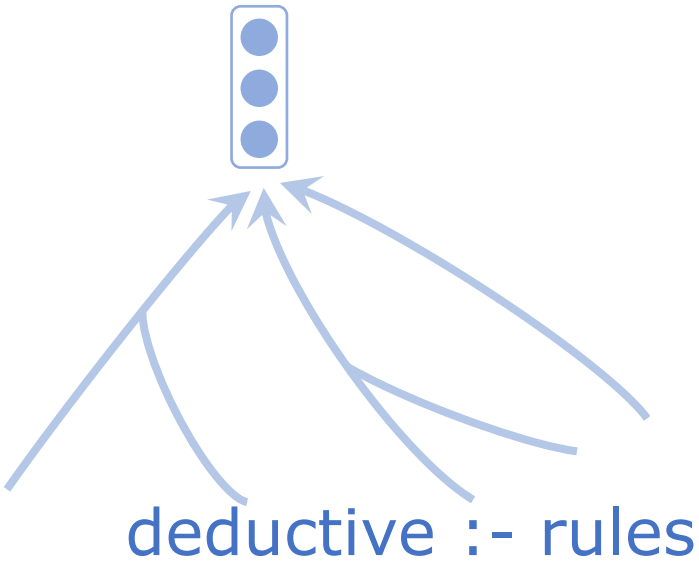


Computing Embeddings & Probabilities

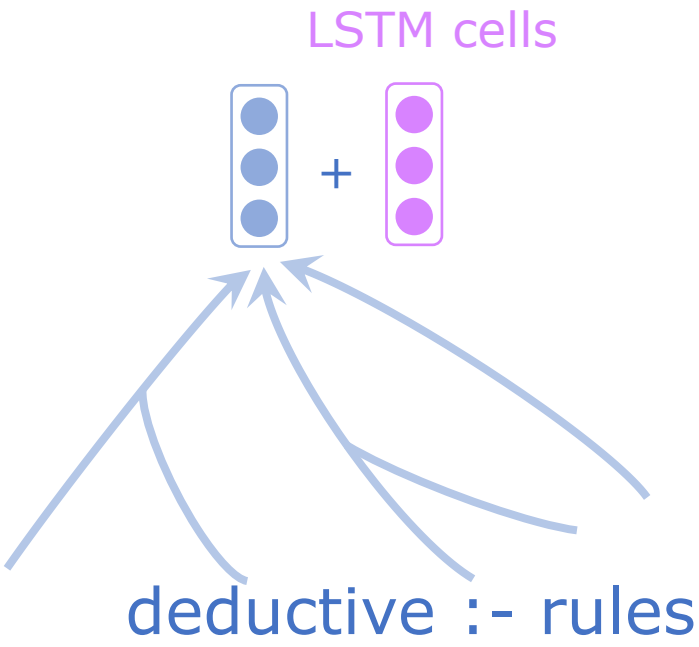


Rules ➔ Deep Recurrent Neural Net

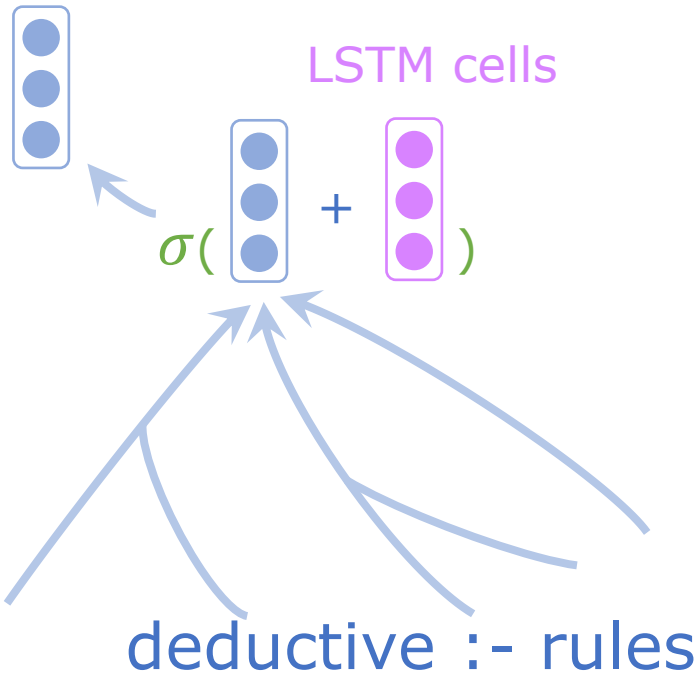
Rules ➔ Deep Recurrent Neural Net



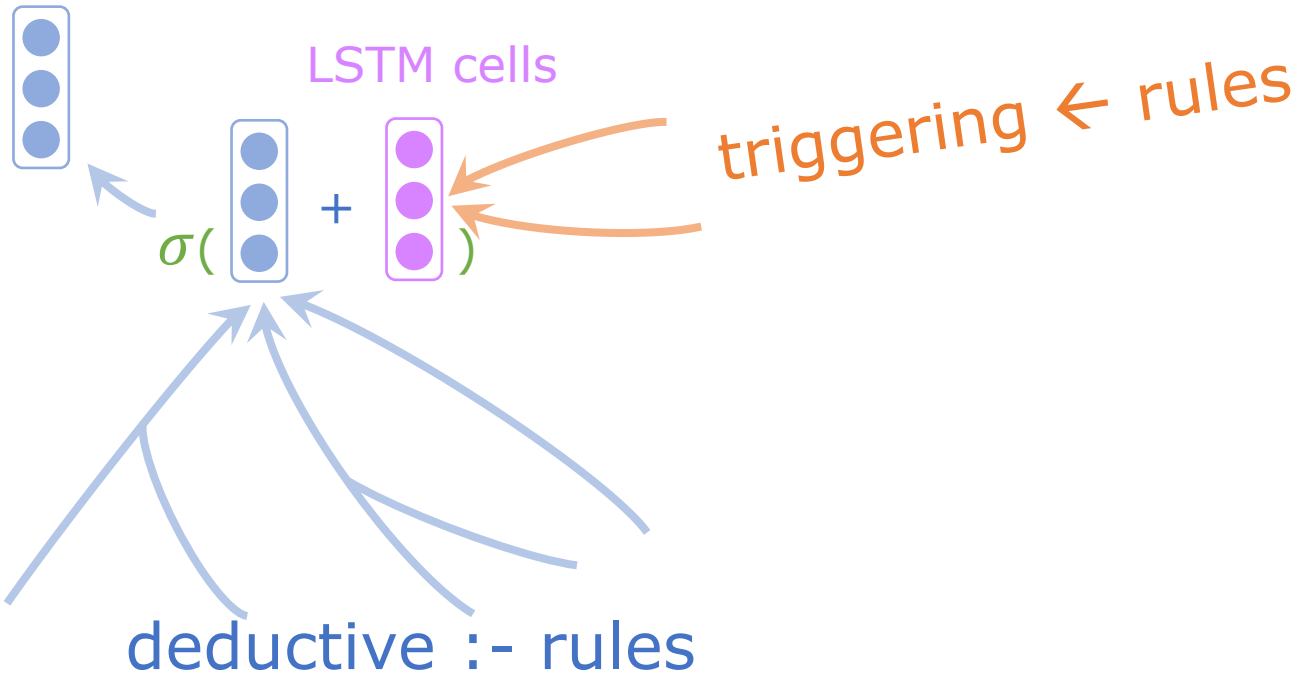
Rules ➔ Deep Recurrent Neural Net



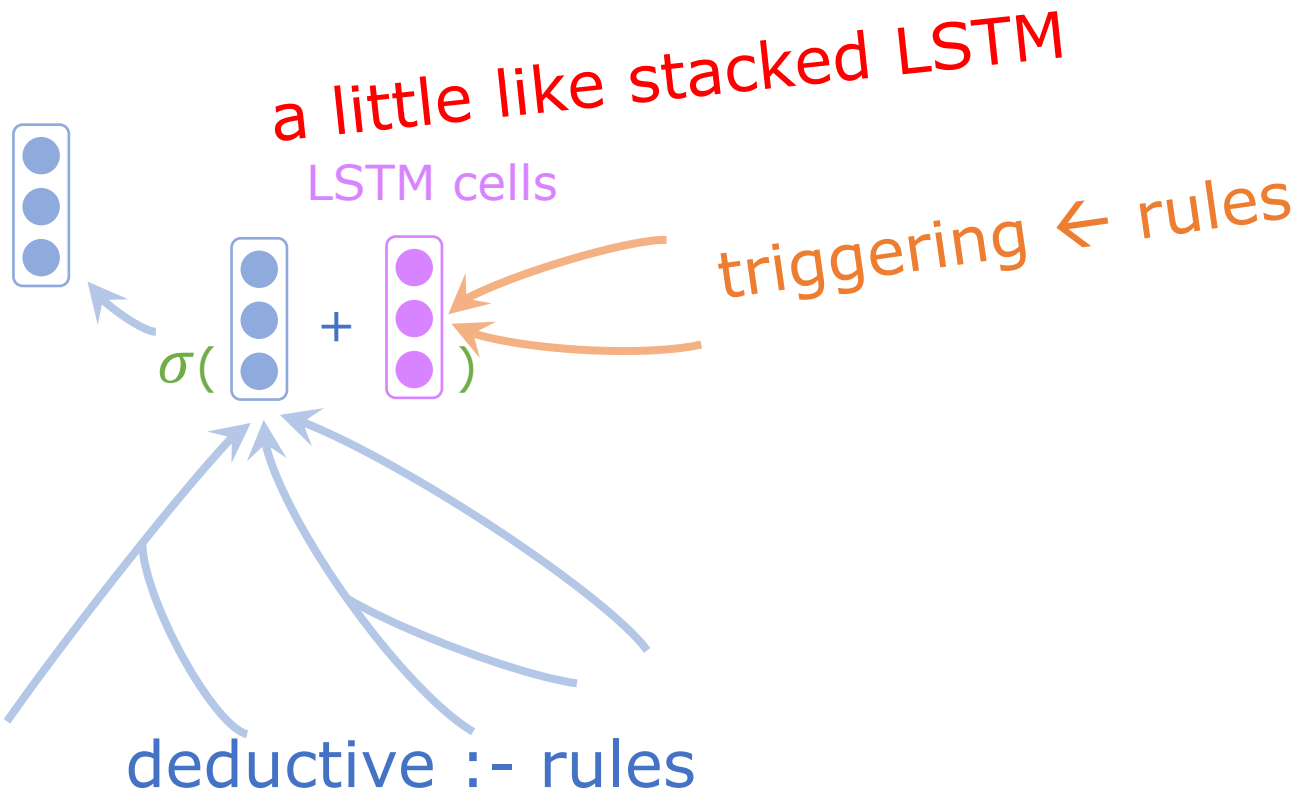
Rules ➔ Deep Recurrent Neural Net



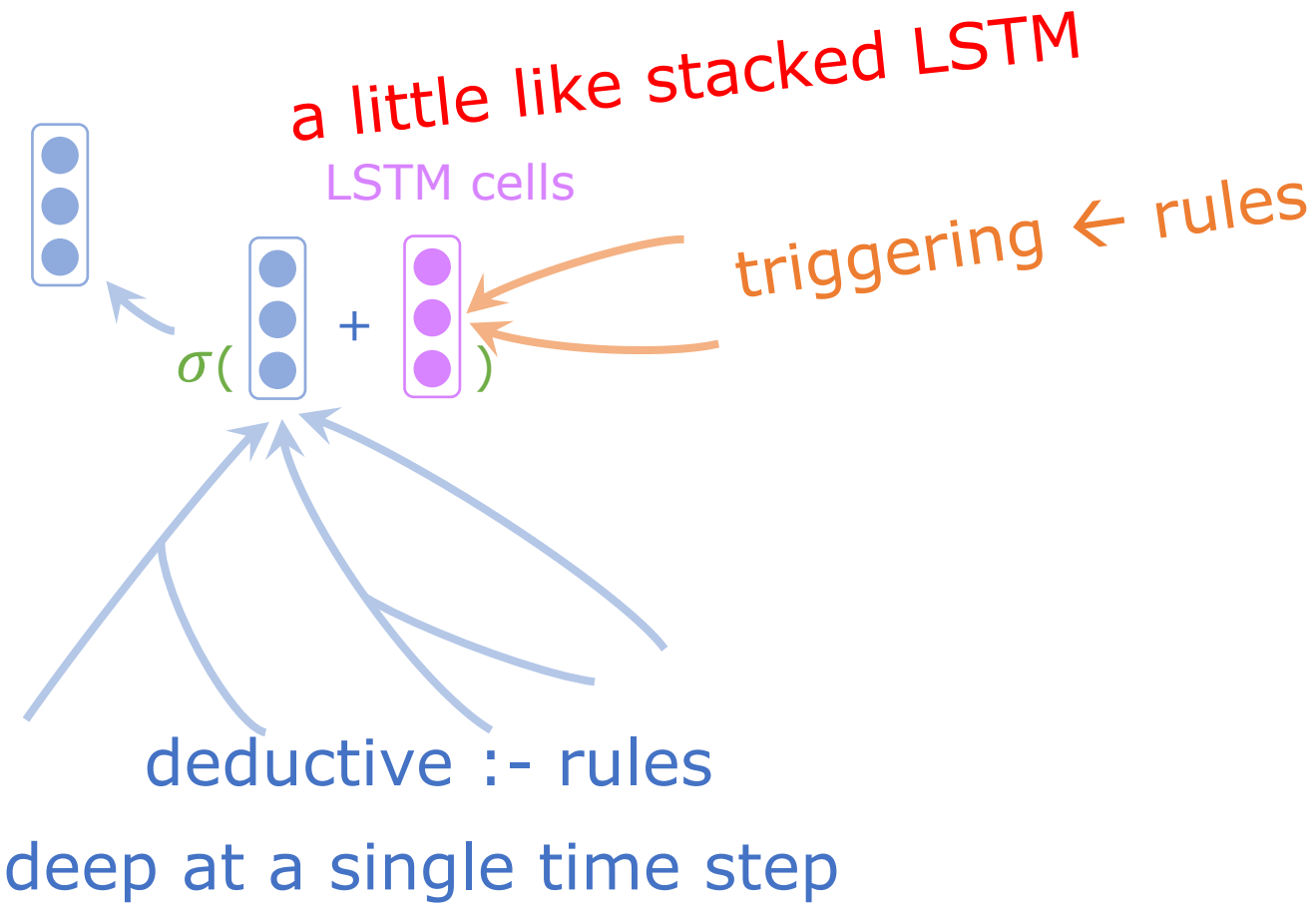
Rules ➔ Deep Recurrent Neural Net



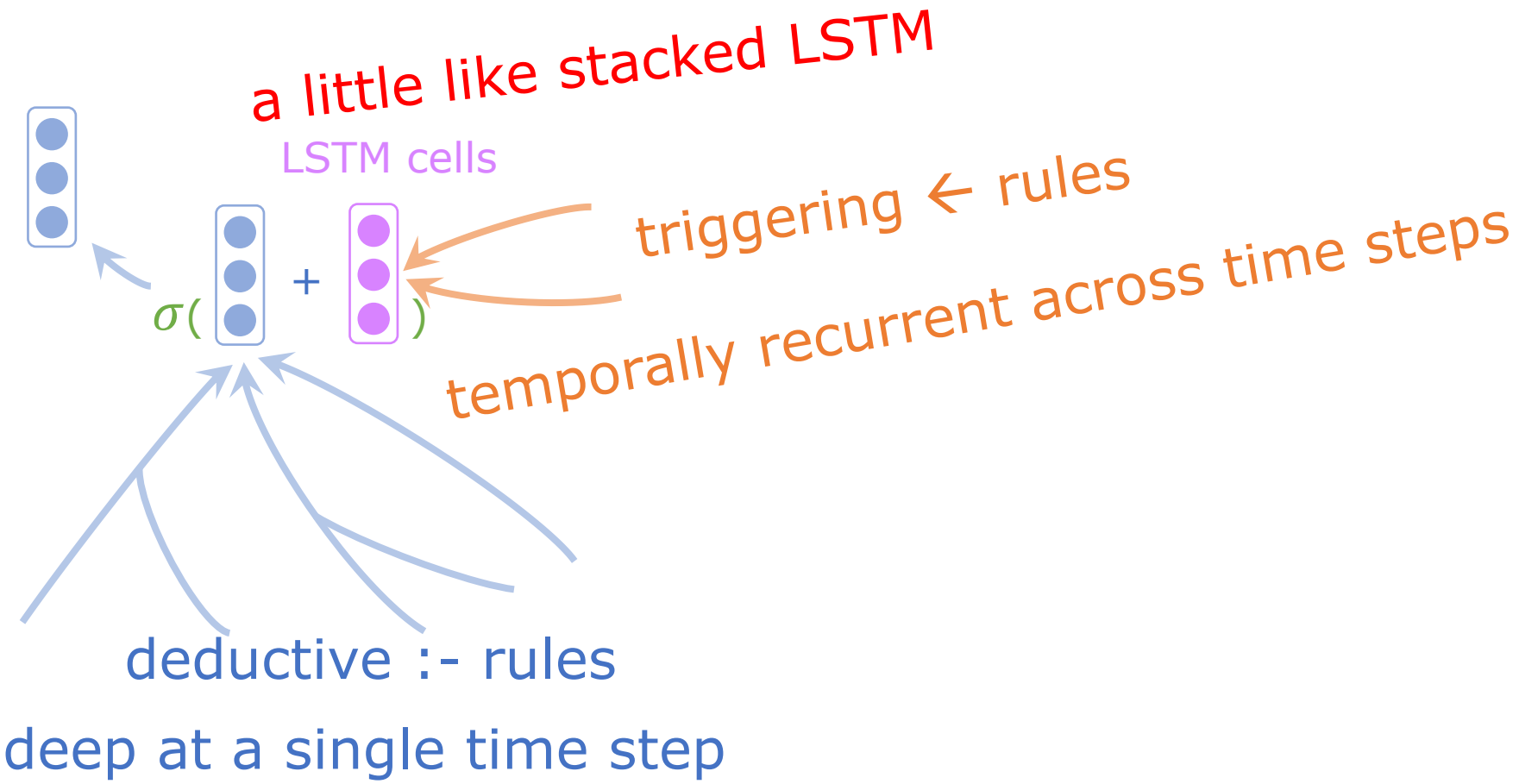
Rules ➔ Deep Recurrent Neural Net



Rules ➔ Deep Recurrent Neural Net



Rules ➔ Deep Recurrent Neural Net



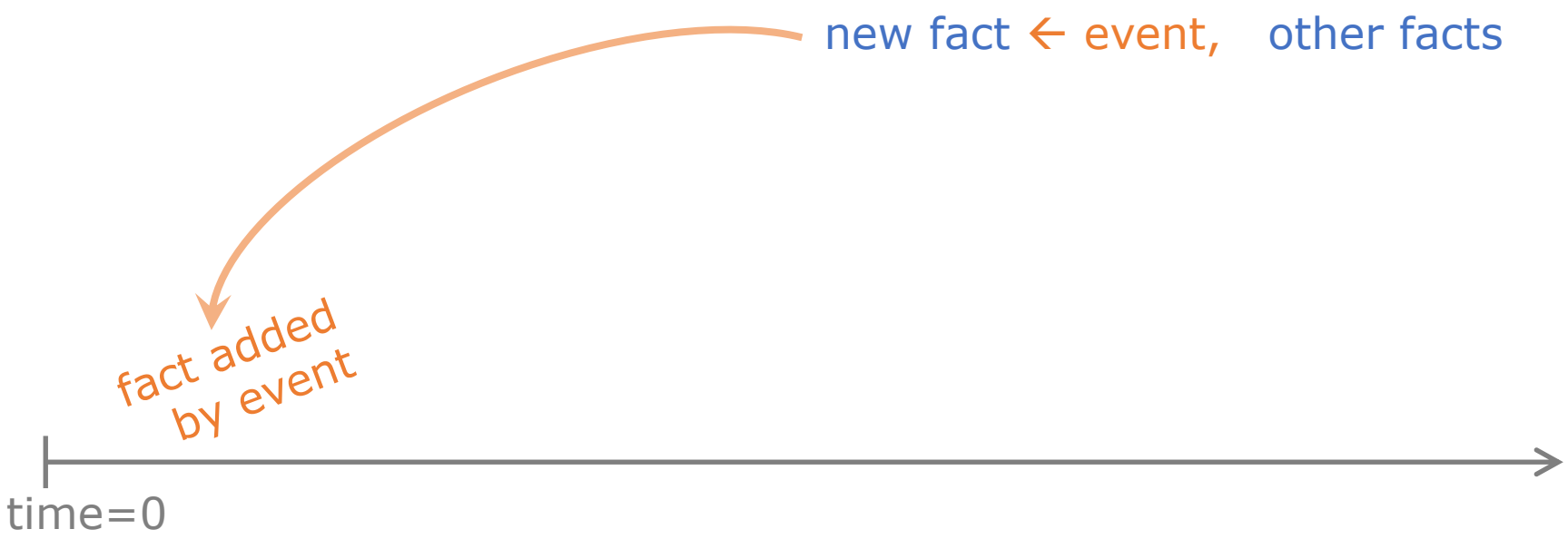
Life Story of a Fact

Life Story of a Fact

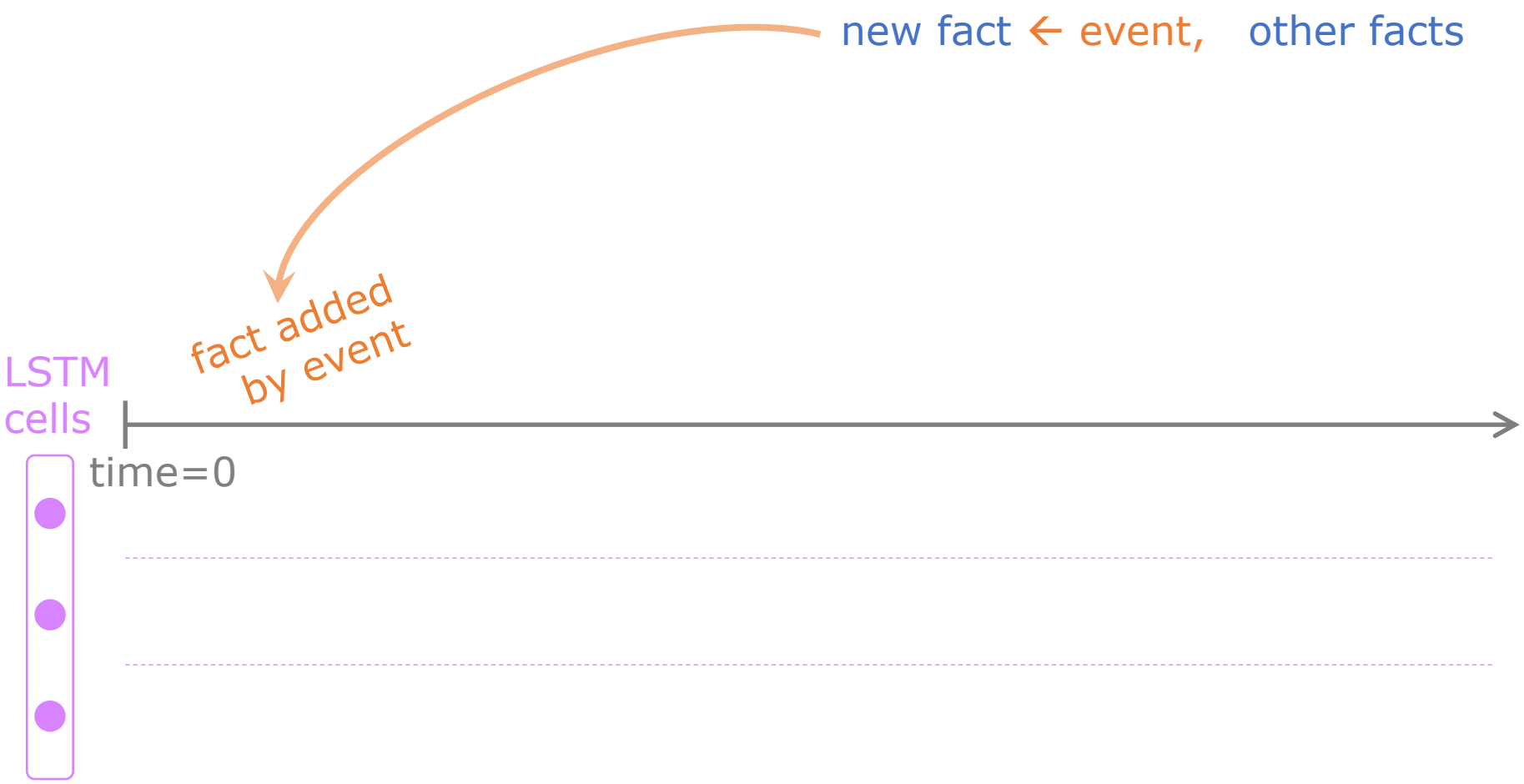
Life Story of a Fact

new fact ← event, other facts

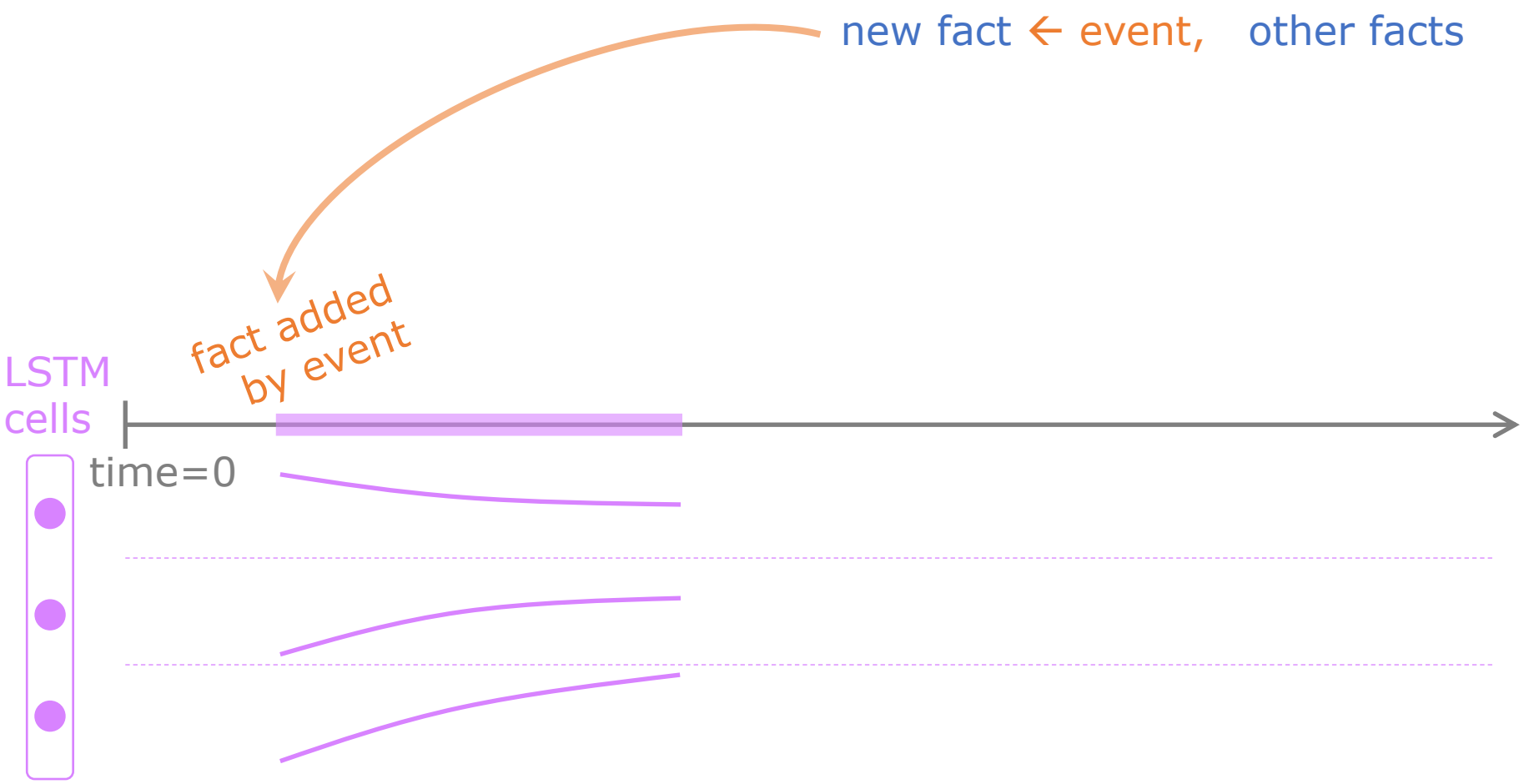
Life Story of a Fact



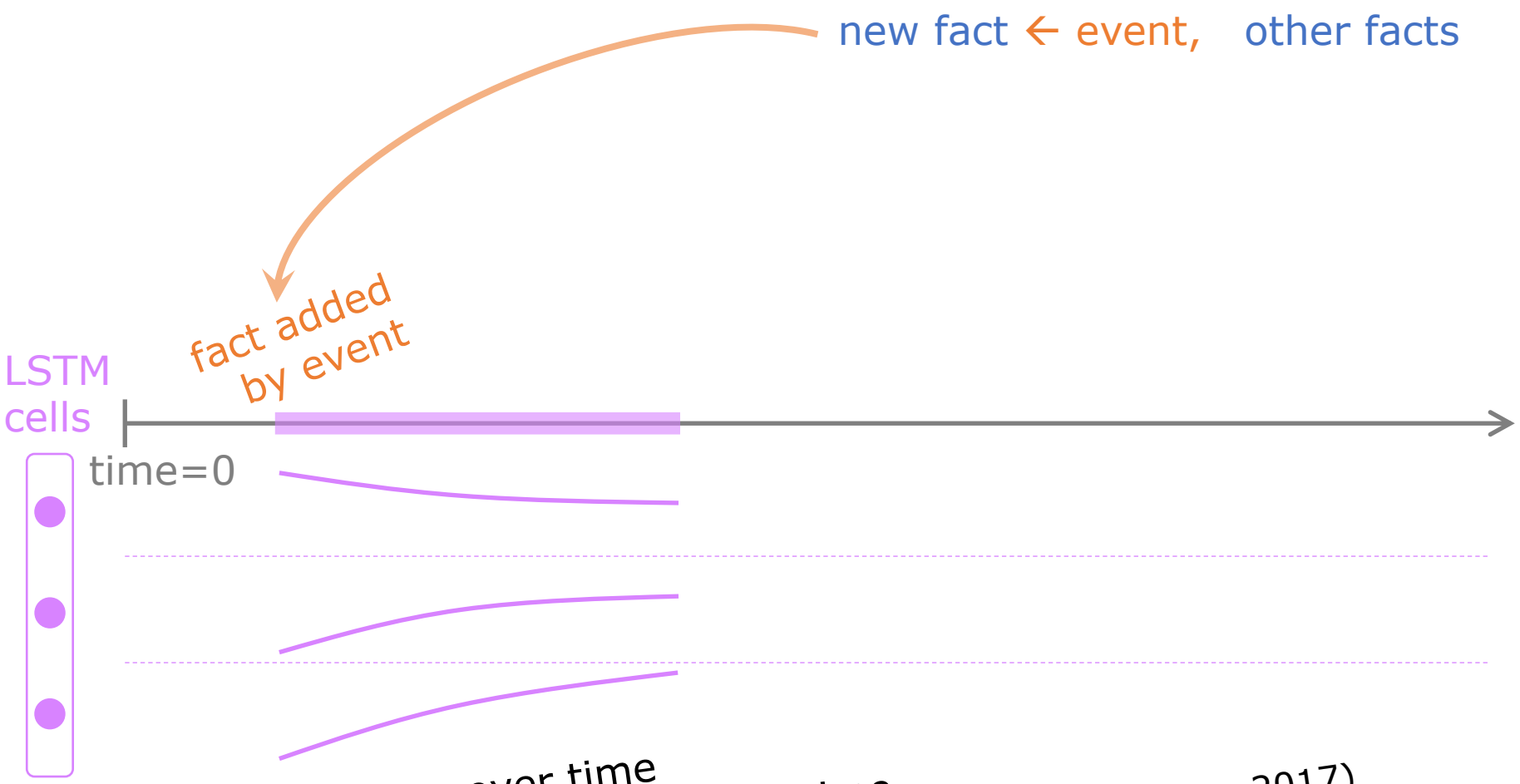
Life Story of a Fact



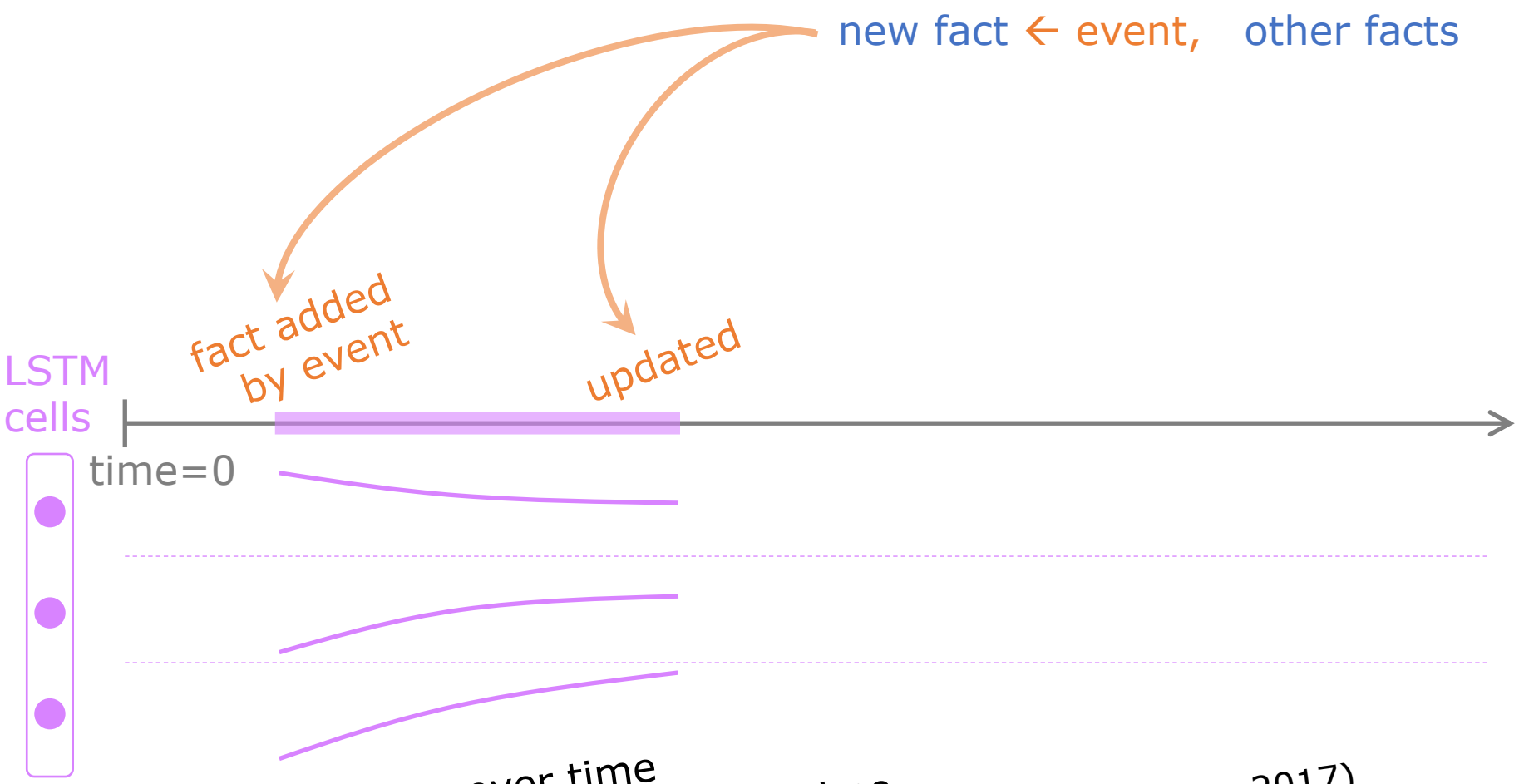
Life Story of a Fact



Life Story of a Fact



Life Story of a Fact

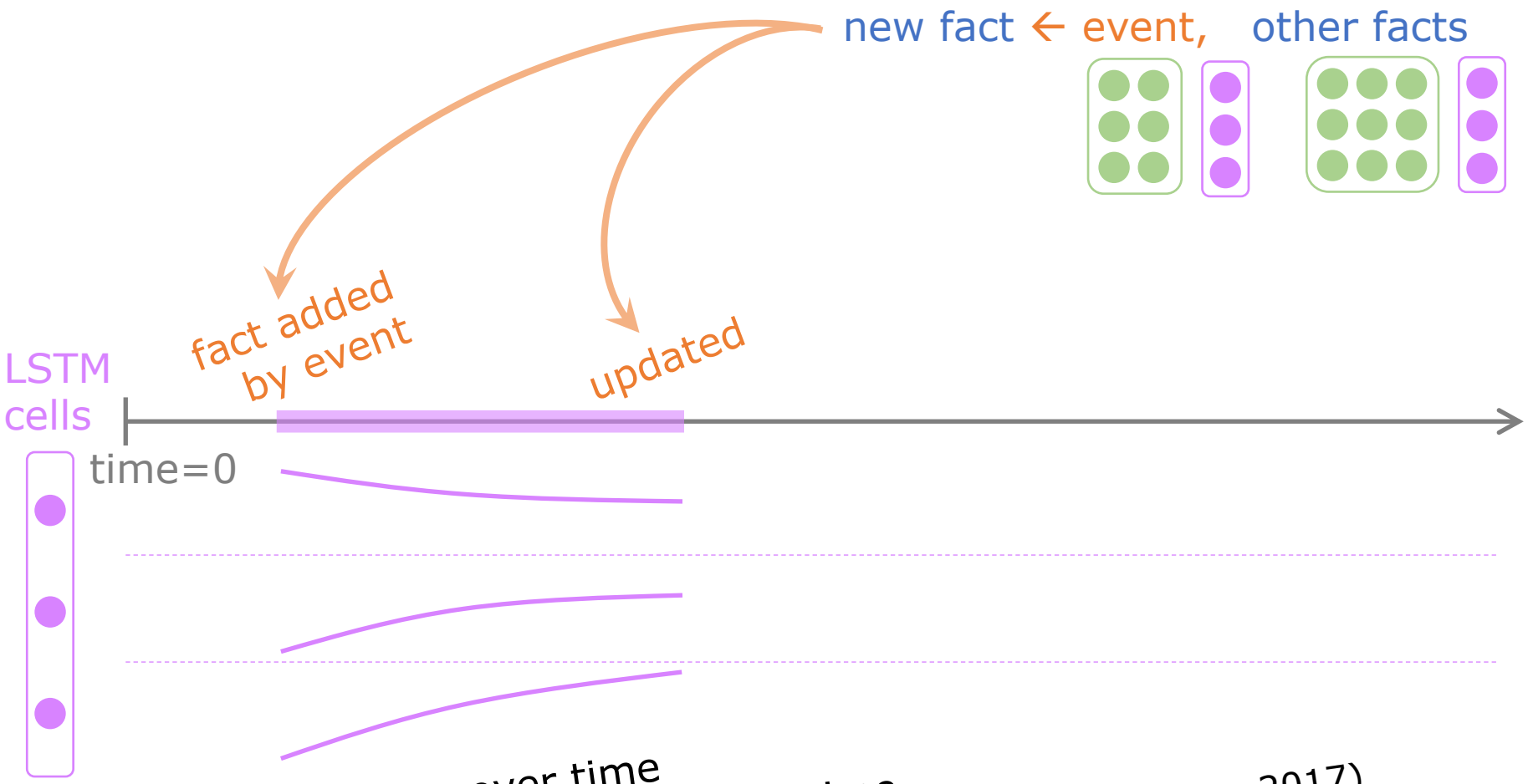


embedding changes over time

probability changes over time

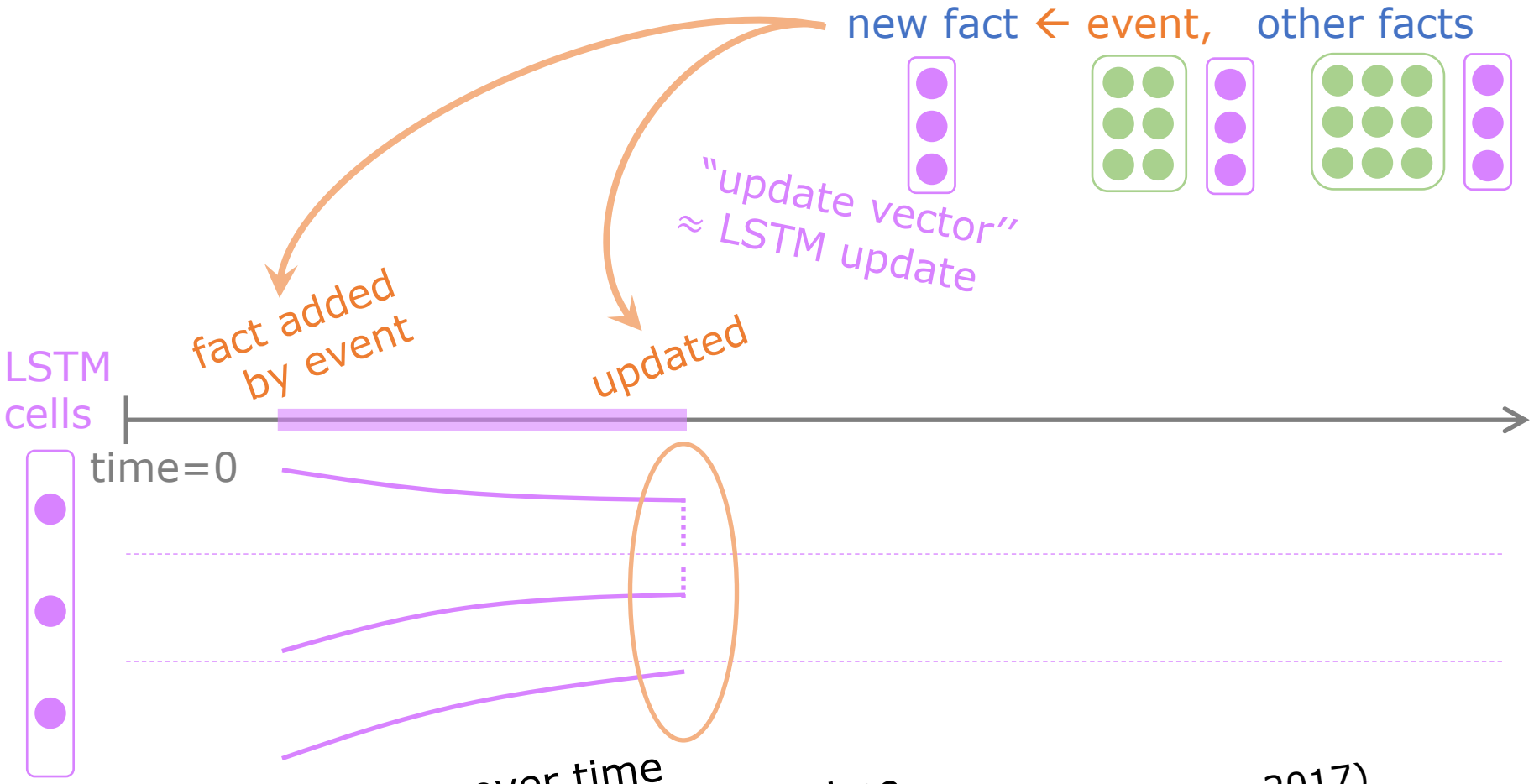
≈ neural Hawkes process (Mei & Eisner, 2017)

Life Story of a Fact



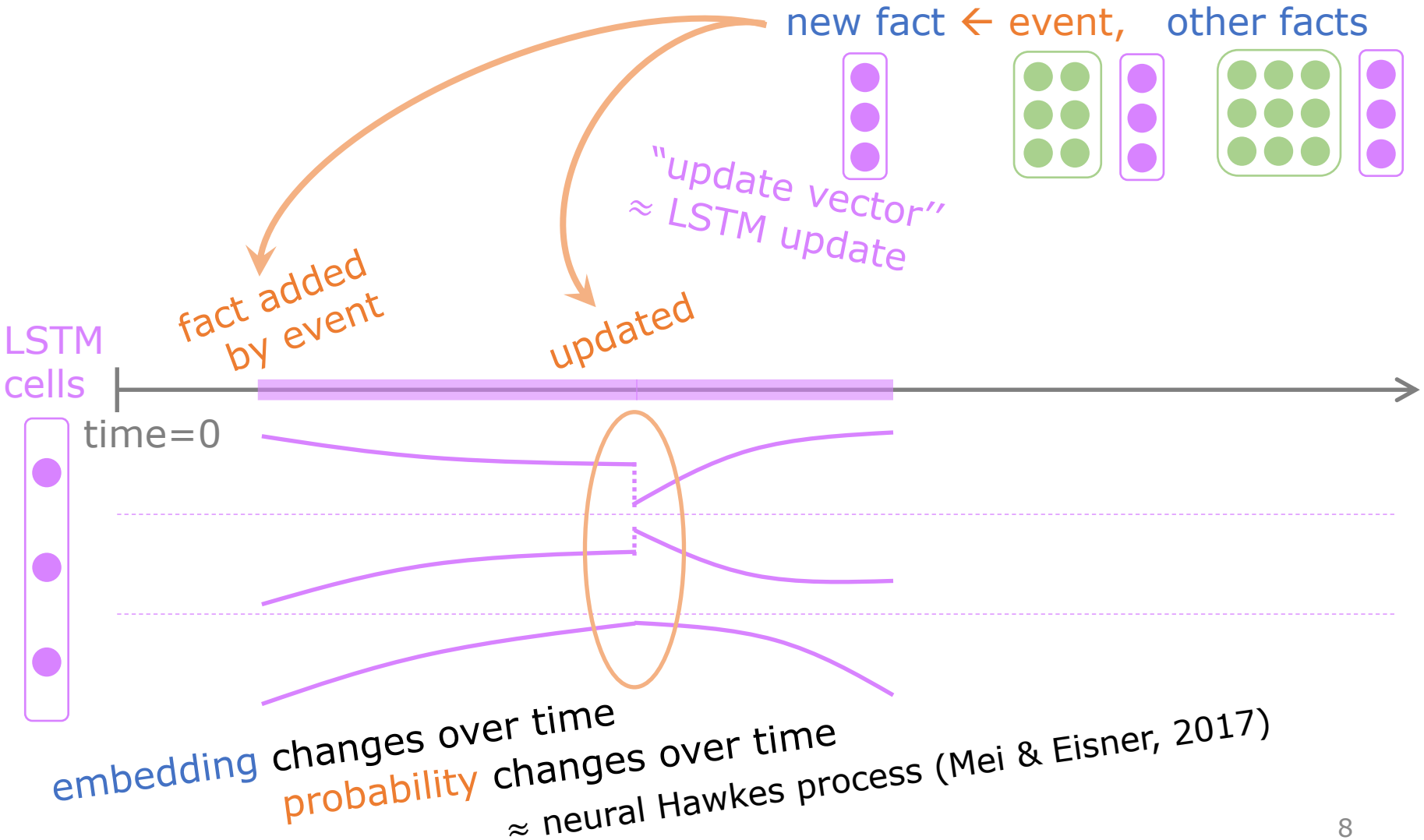
embedding changes over time
probability changes over time
≈ neural Hawkes process (Mei & Eisner, 2017)

Life Story of a Fact

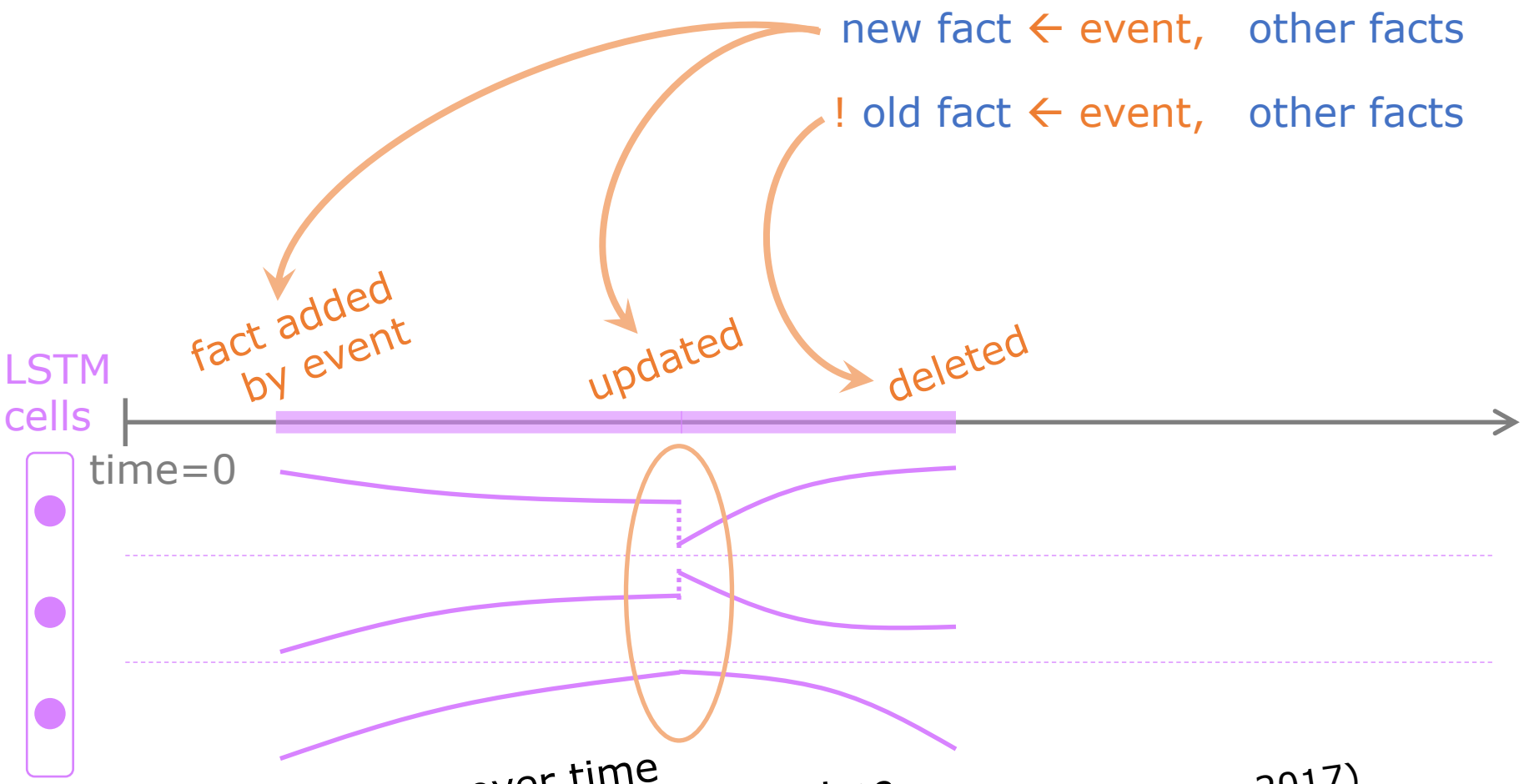


embedding changes over time
probability changes over time
≈ neural Hawkes process (Mei & Eisner, 2017)

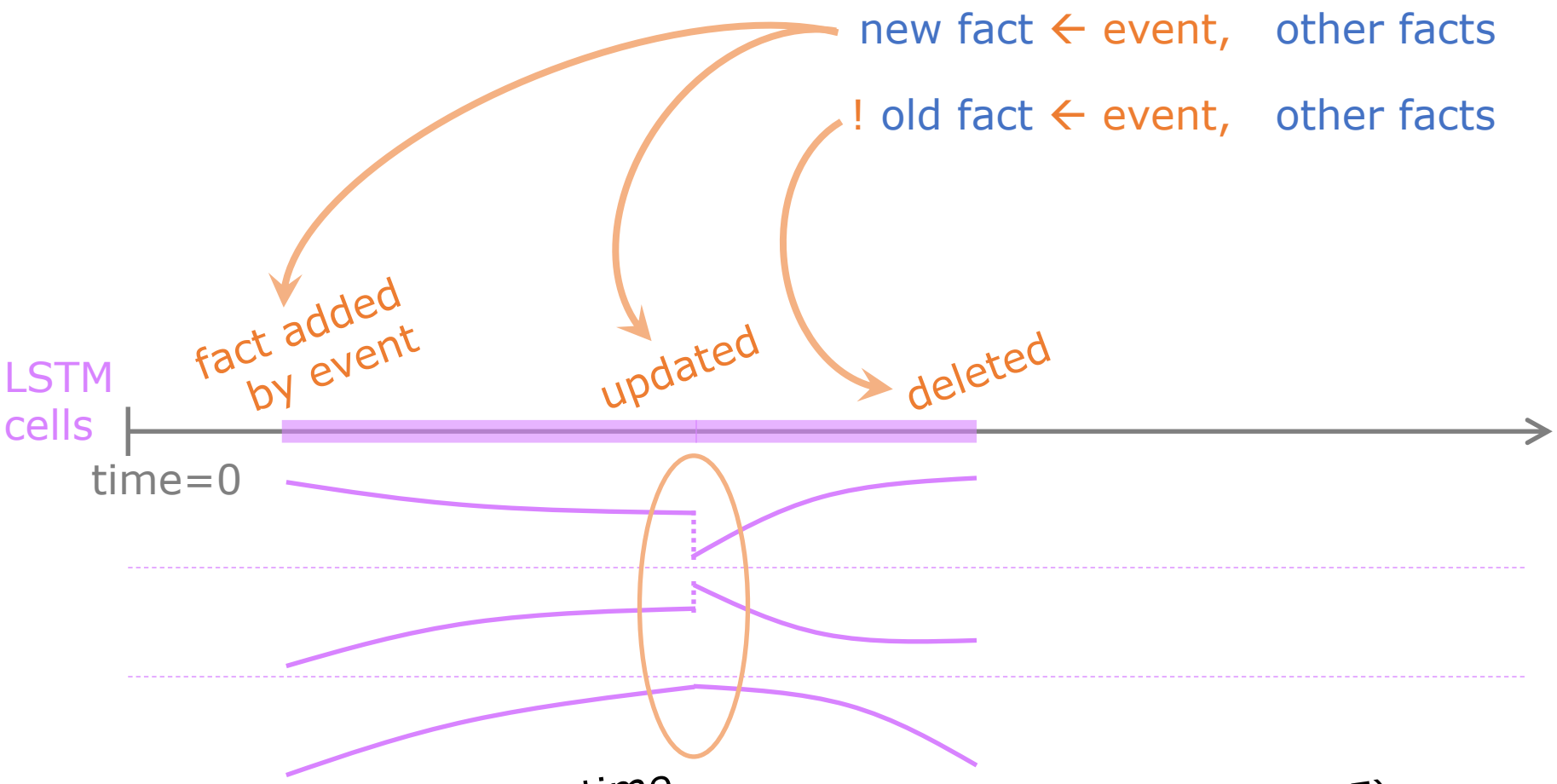
Life Story of a Fact



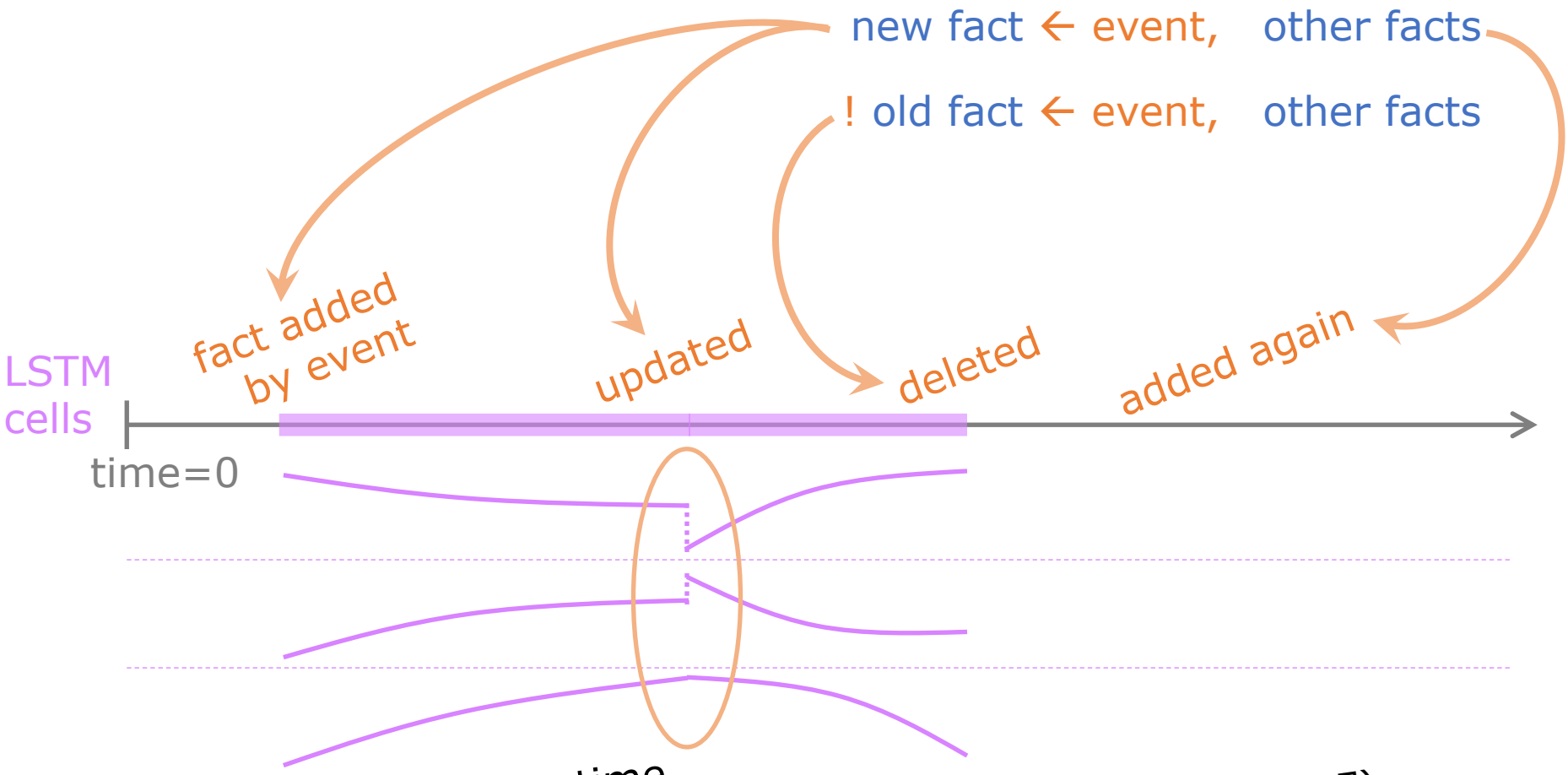
Life Story of a Fact



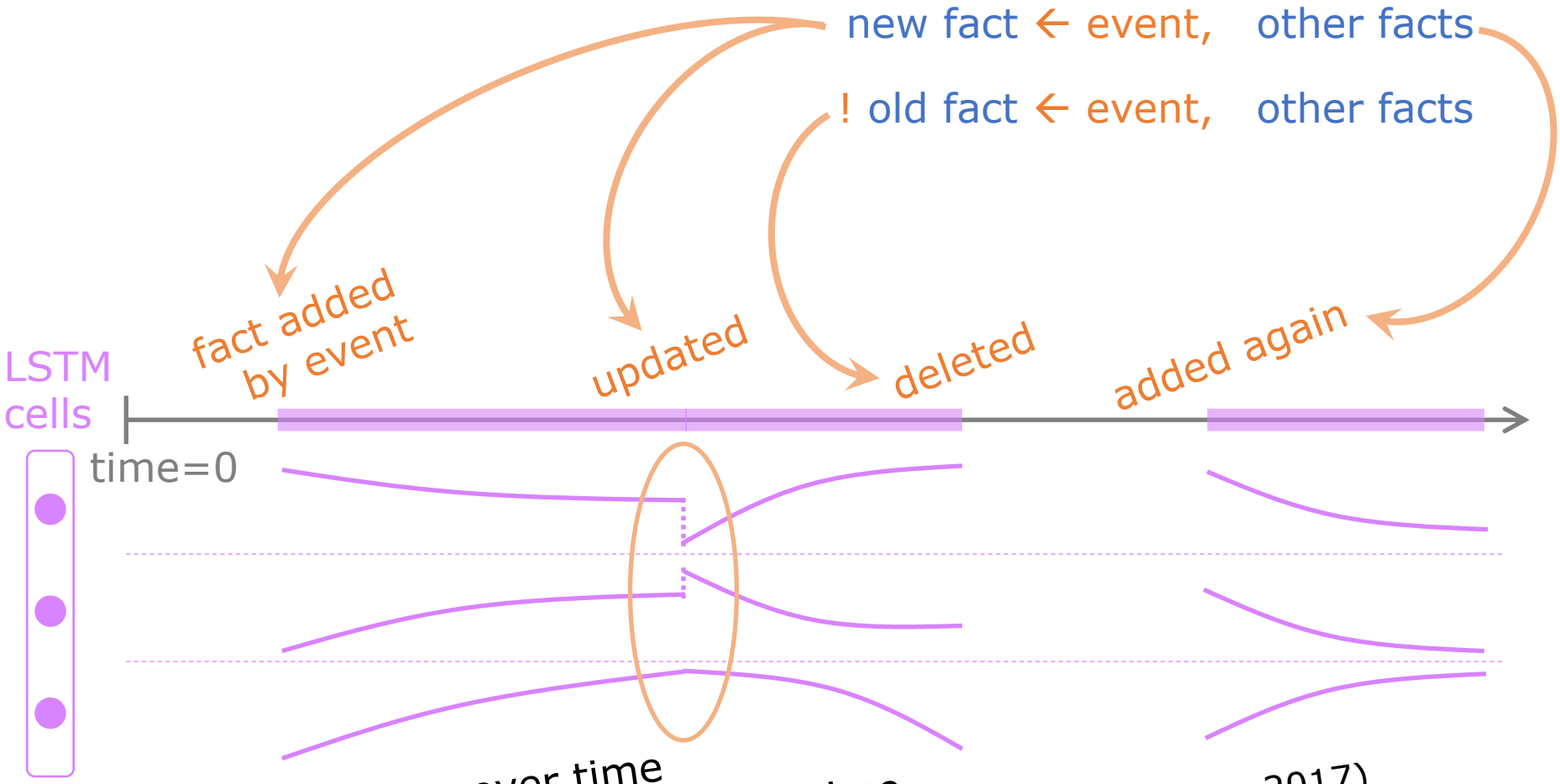
Life Story of a Fact



Life Story of a Fact



Life Story of a Fact



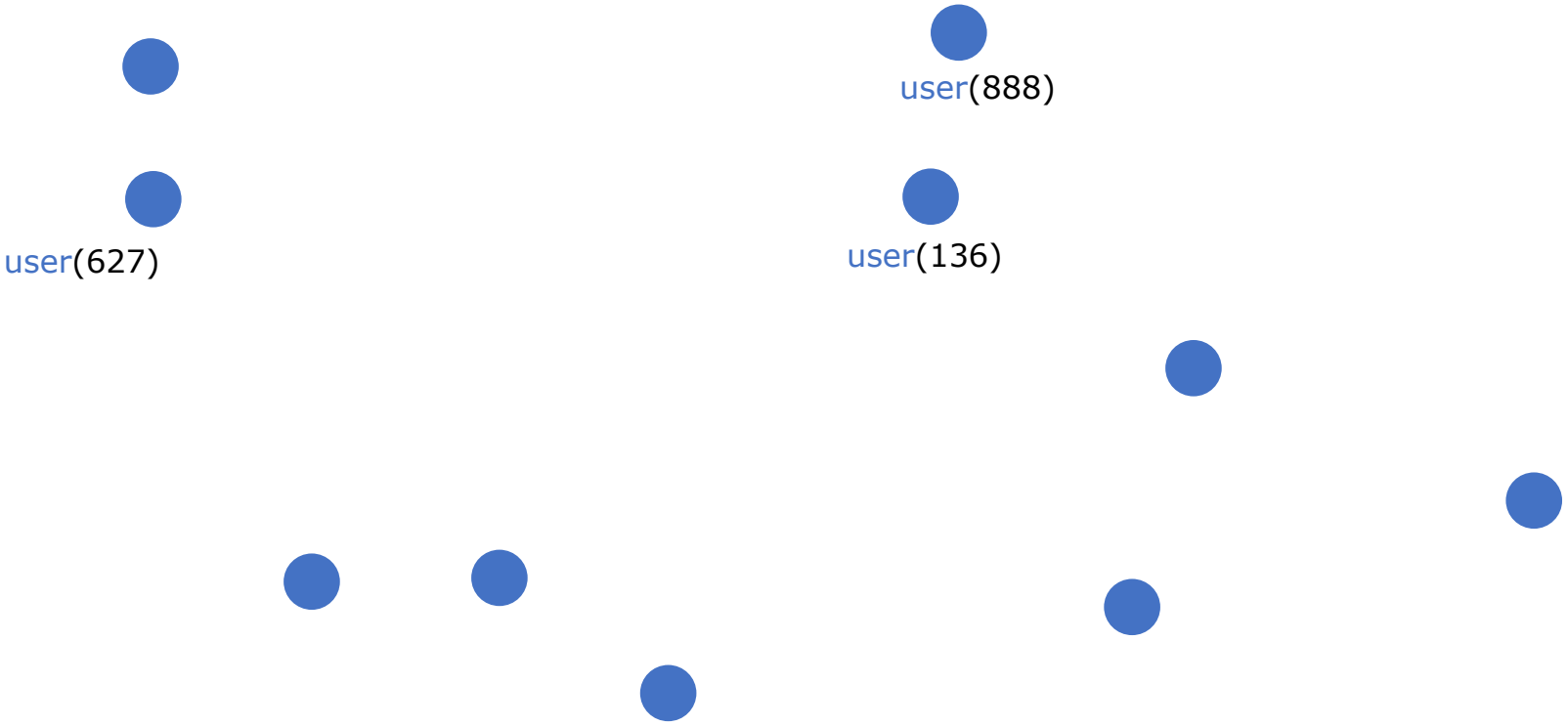
Experiment: Users watch TV programs

collaborative filtering problem with timing
who watches what and when?

Experiment: Users watch TV programs

collaborative filtering problem with timing
who watches what and when?

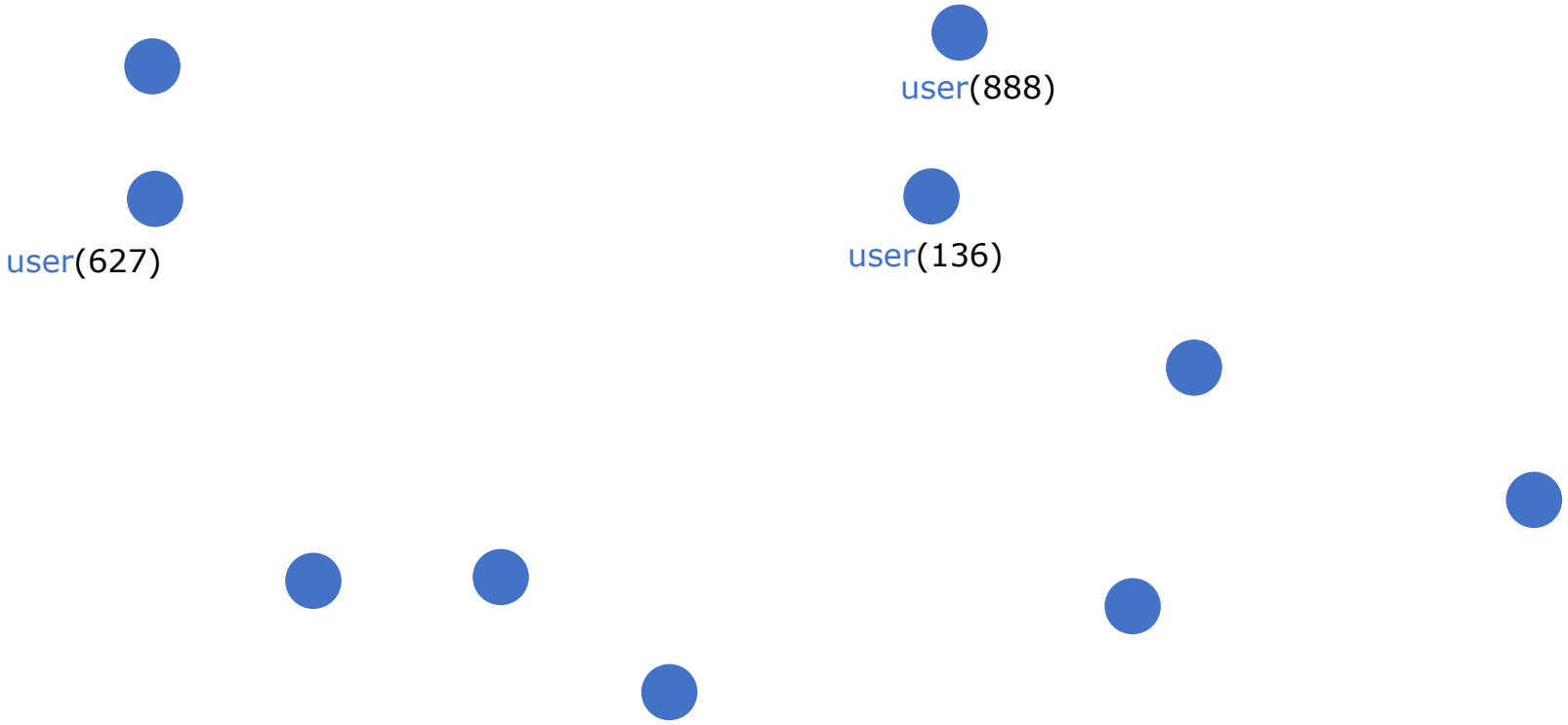
1000 users



Experiment: Users watch TV programs

collaborative filtering problem with timing
who watches what and when?

1000 users 49 TV programs to be released

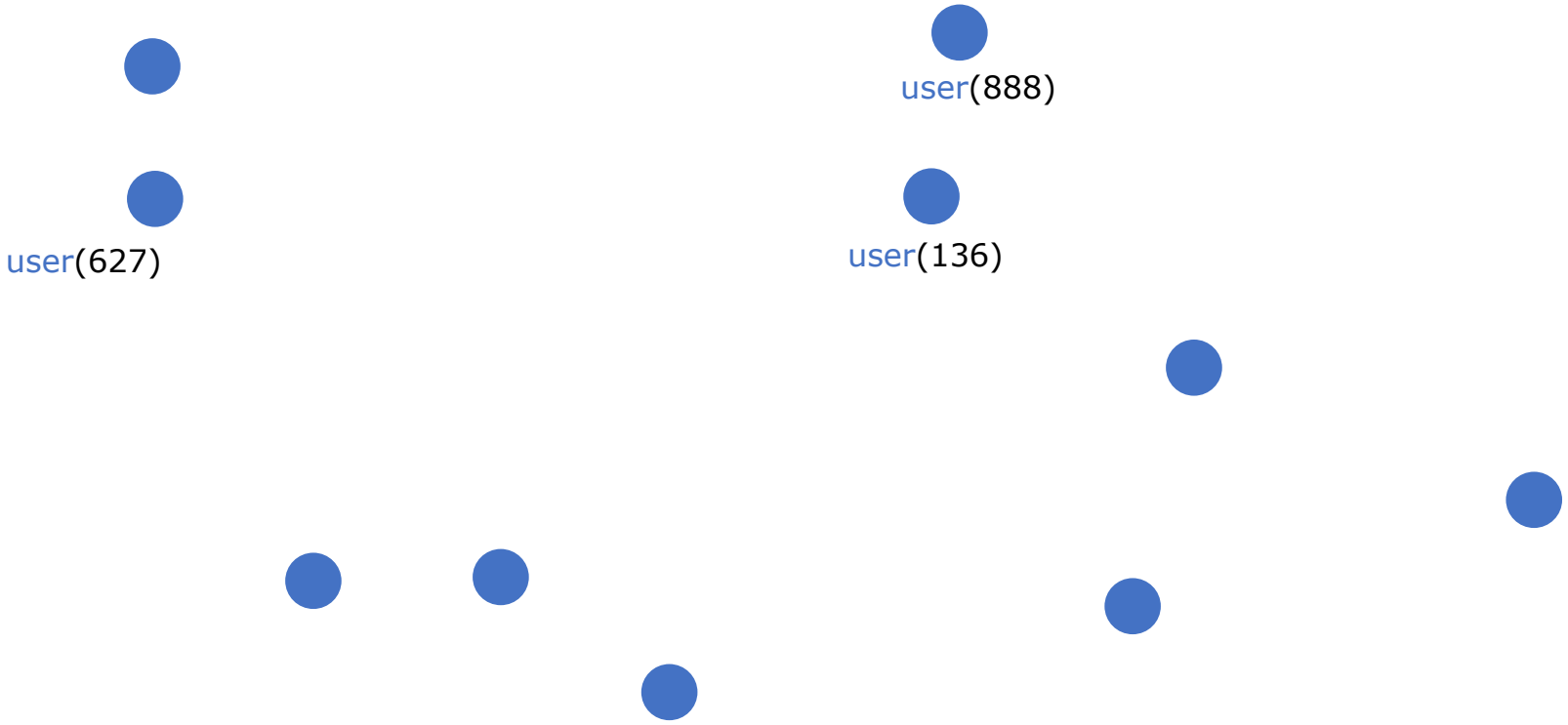


Experiment: Users watch TV programs

collaborative filtering problem with timing
who watches what and when?

1000 users 49 TV programs to be released

49000 possible watch events

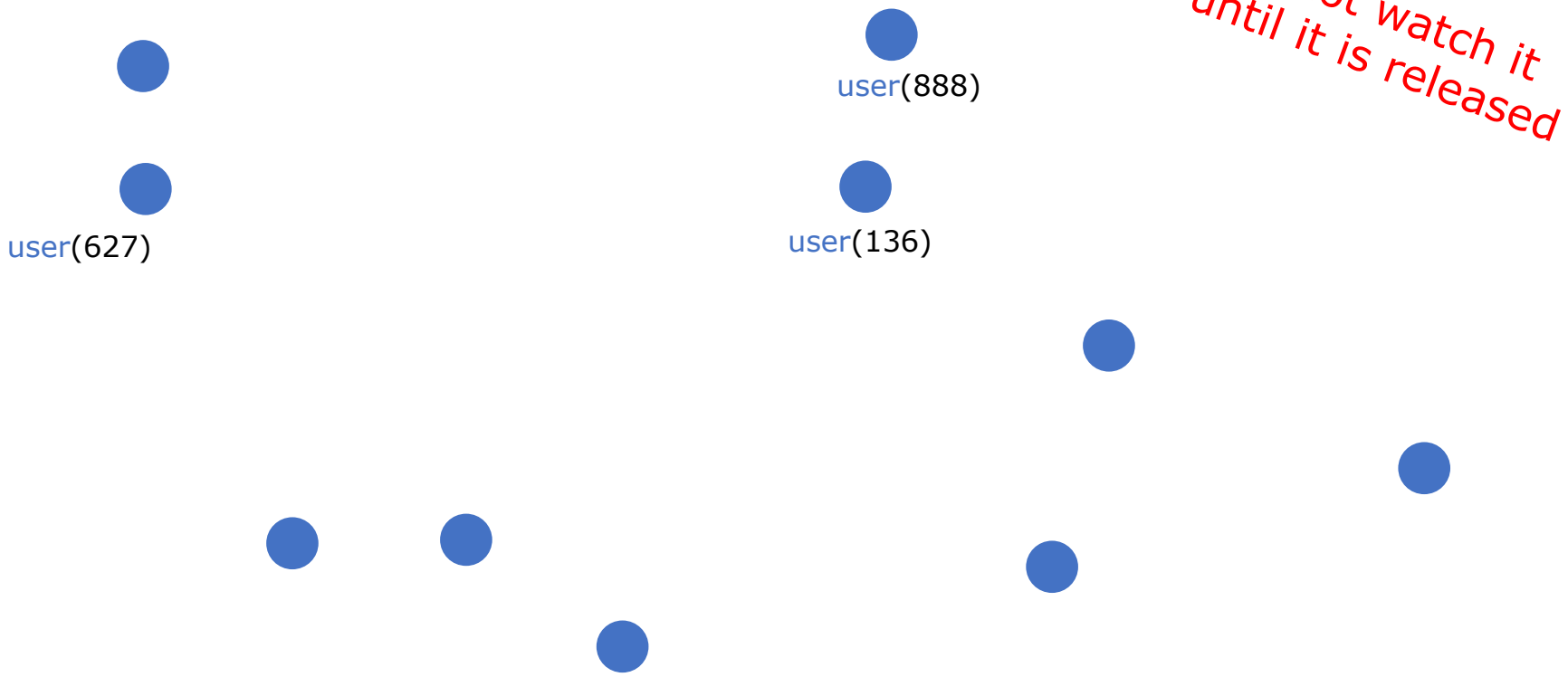


Experiment: Users watch TV programs

collaborative filtering problem with timing
who watches what and when?

1000 users 49 TV programs to be released

49000 possible watch events



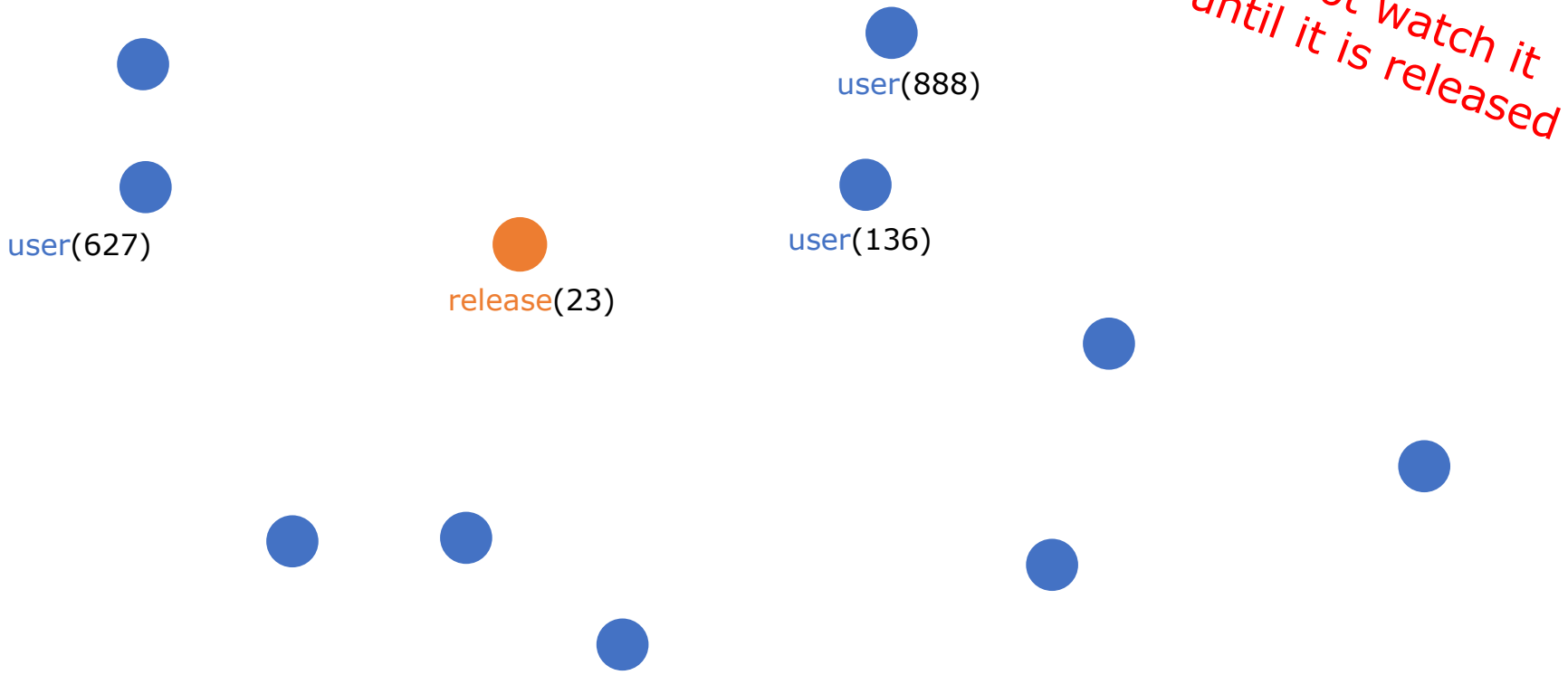
can not watch it
until it is released

Experiment: Users watch TV programs

collaborative filtering problem with timing
who watches what and when?

1000 users 49 TV programs to be released

49000 possible watch events

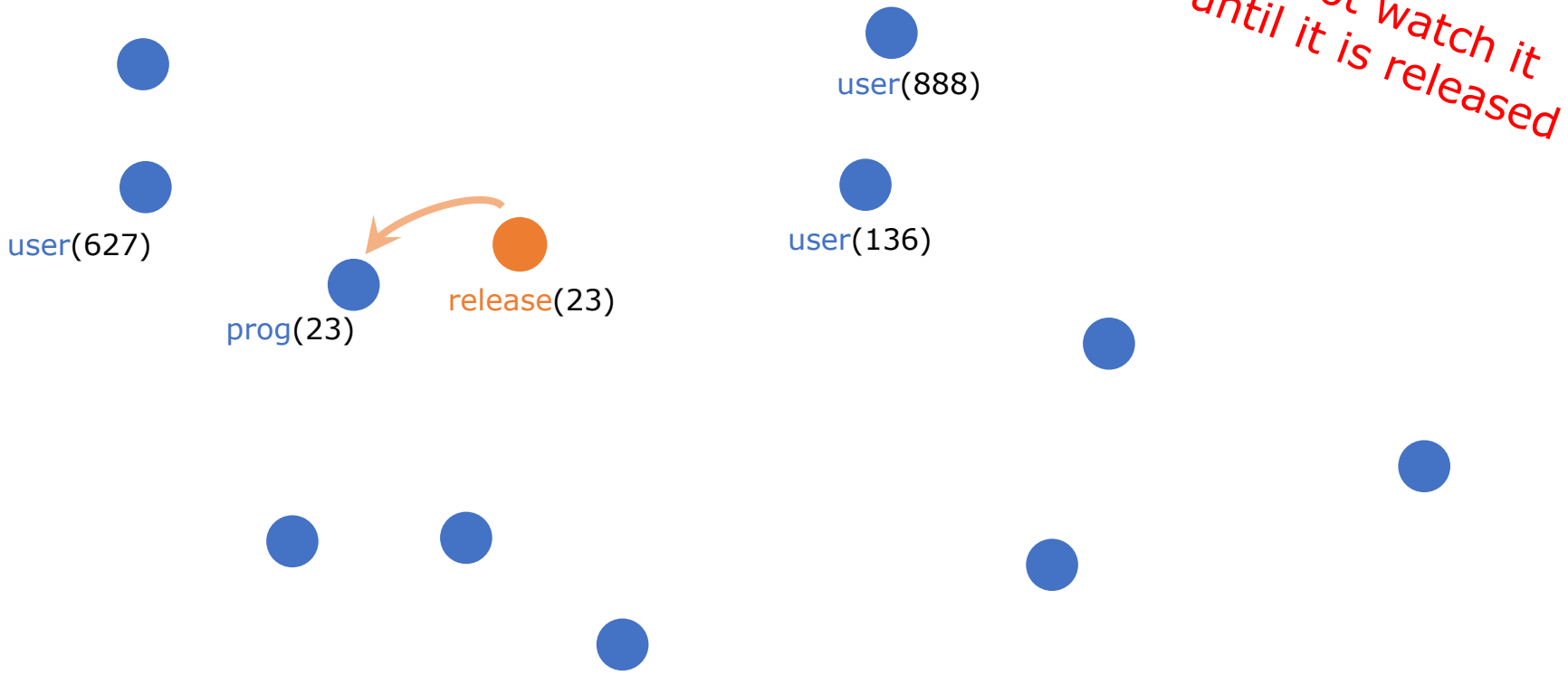


Experiment: Users watch TV programs

collaborative filtering problem with timing
who watches what and when?

1000 users 49 TV programs to be released

49000 possible watch events



can not watch it
until it is released

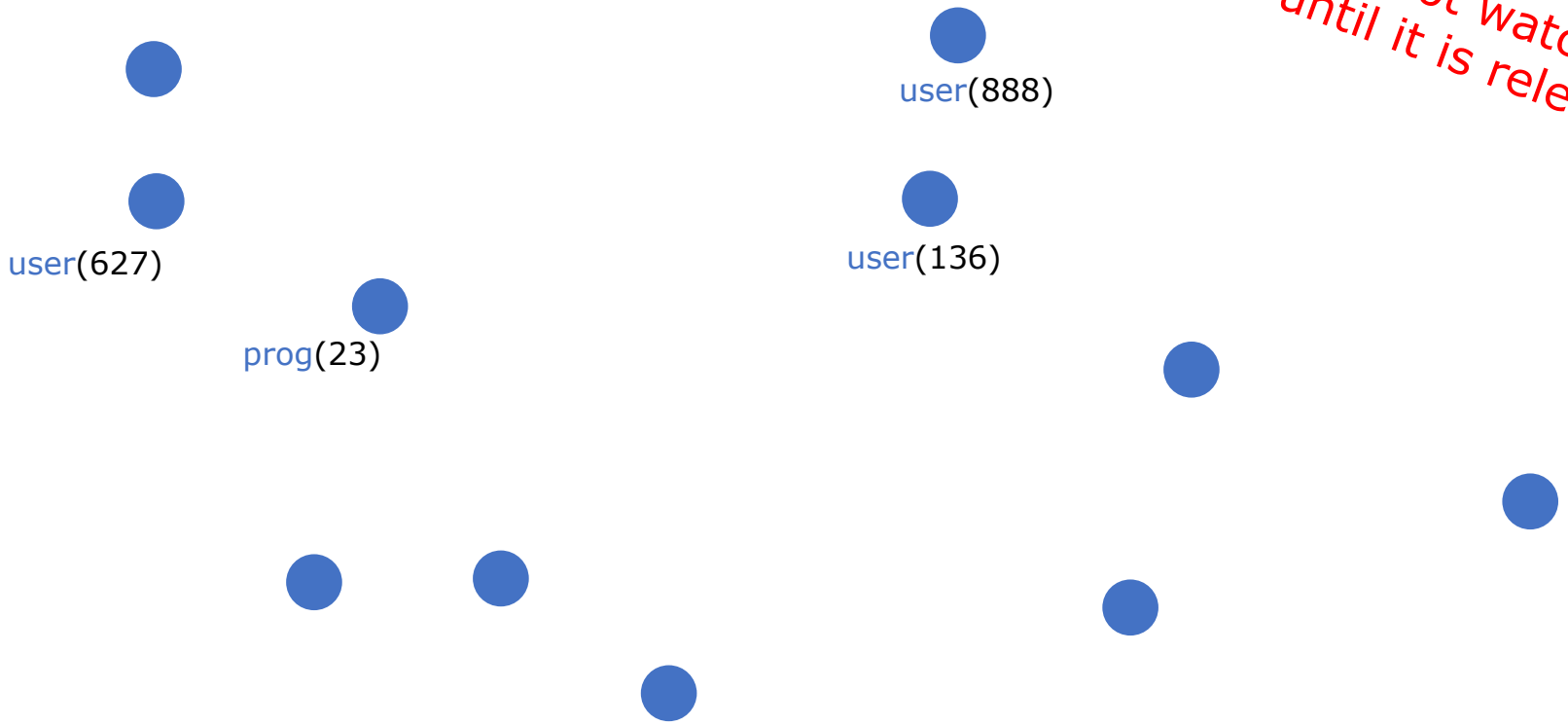
Experiment: Users watch TV programs

collaborative filtering problem with timing
who watches what and when?

1000 users 49 TV programs to be released

49000 possible watch events

can not watch it
until it is released

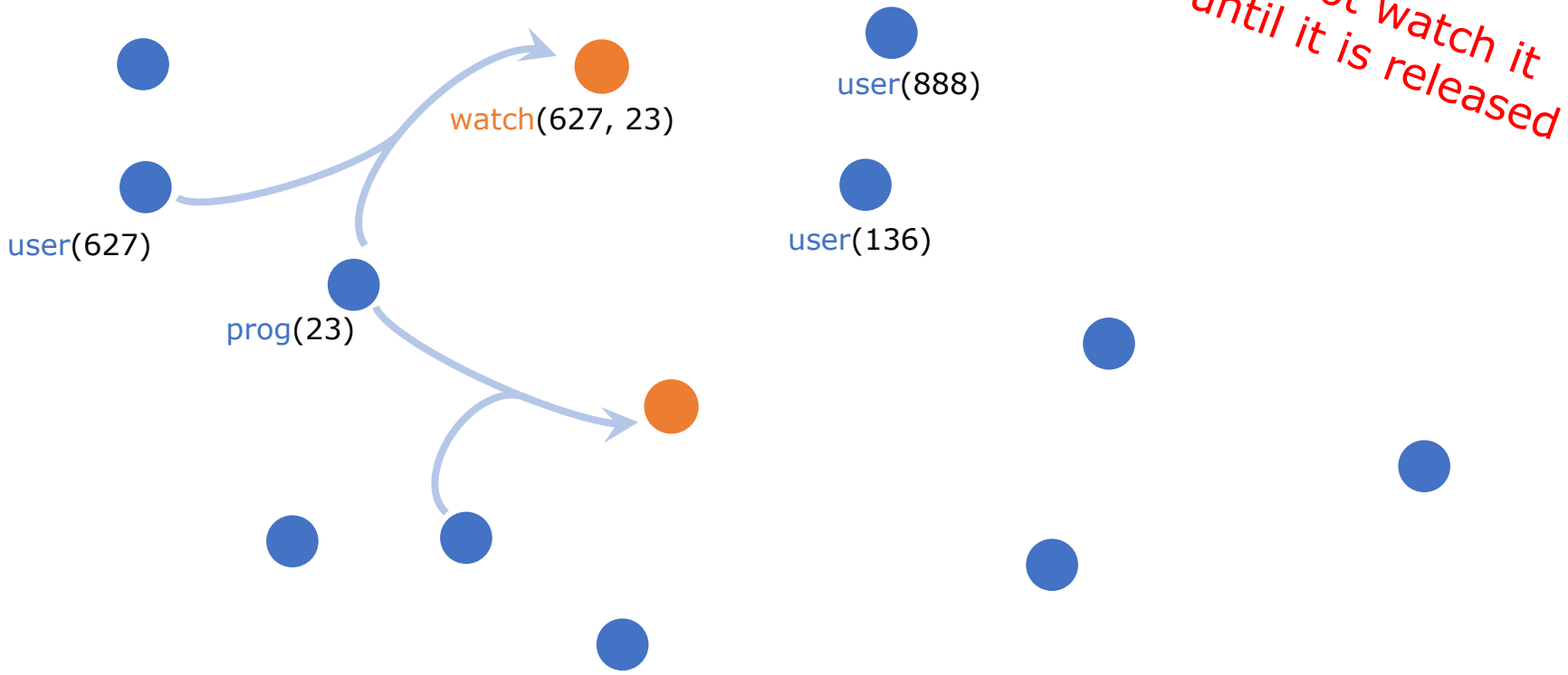


Experiment: Users watch TV programs

collaborative filtering problem with timing
who watches what and when?

1000 users 49 TV programs to be released

49000 possible watch events



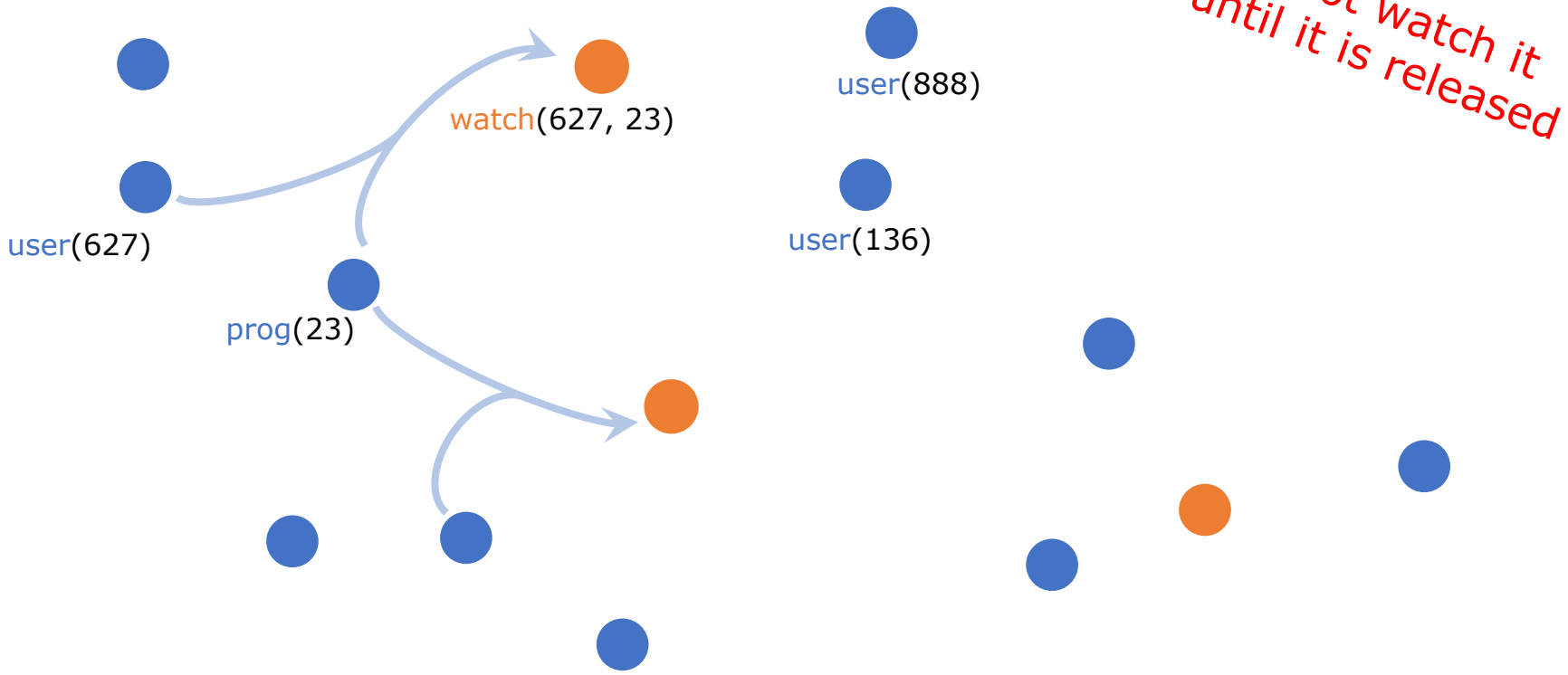
can not watch it
until it is released

Experiment: Users watch TV programs

collaborative filtering problem with timing
who watches what and when?

1000 users 49 TV programs to be released

49000 possible watch events



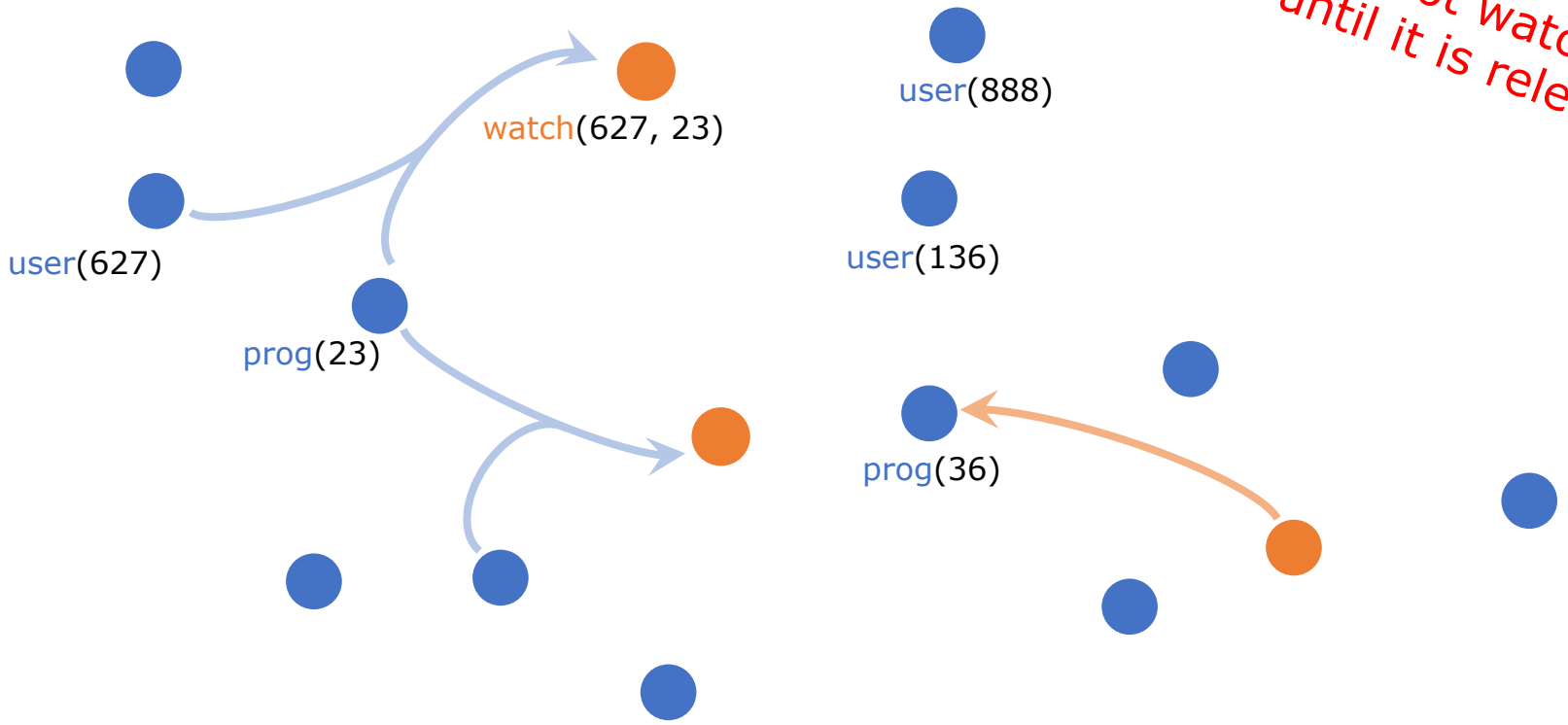
can not watch it
until it is released

Experiment: Users watch TV programs

collaborative filtering problem with timing
who watches what and when?

1000 users 49 TV programs to be released

49000 possible watch events



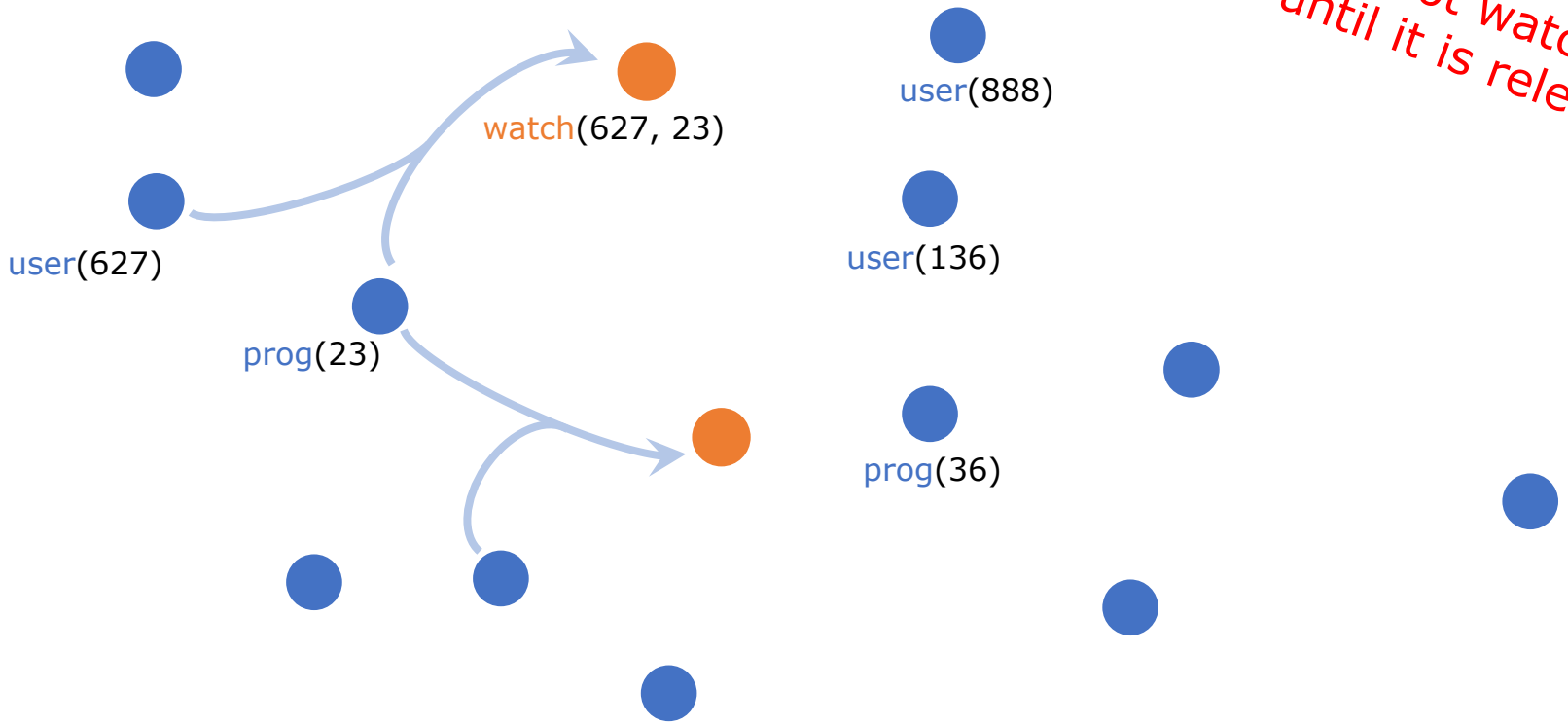
can not watch it
until it is released

Experiment: Users watch TV programs

collaborative filtering problem with timing
who watches what and when?

1000 users 49 TV programs to be released

49000 possible watch events



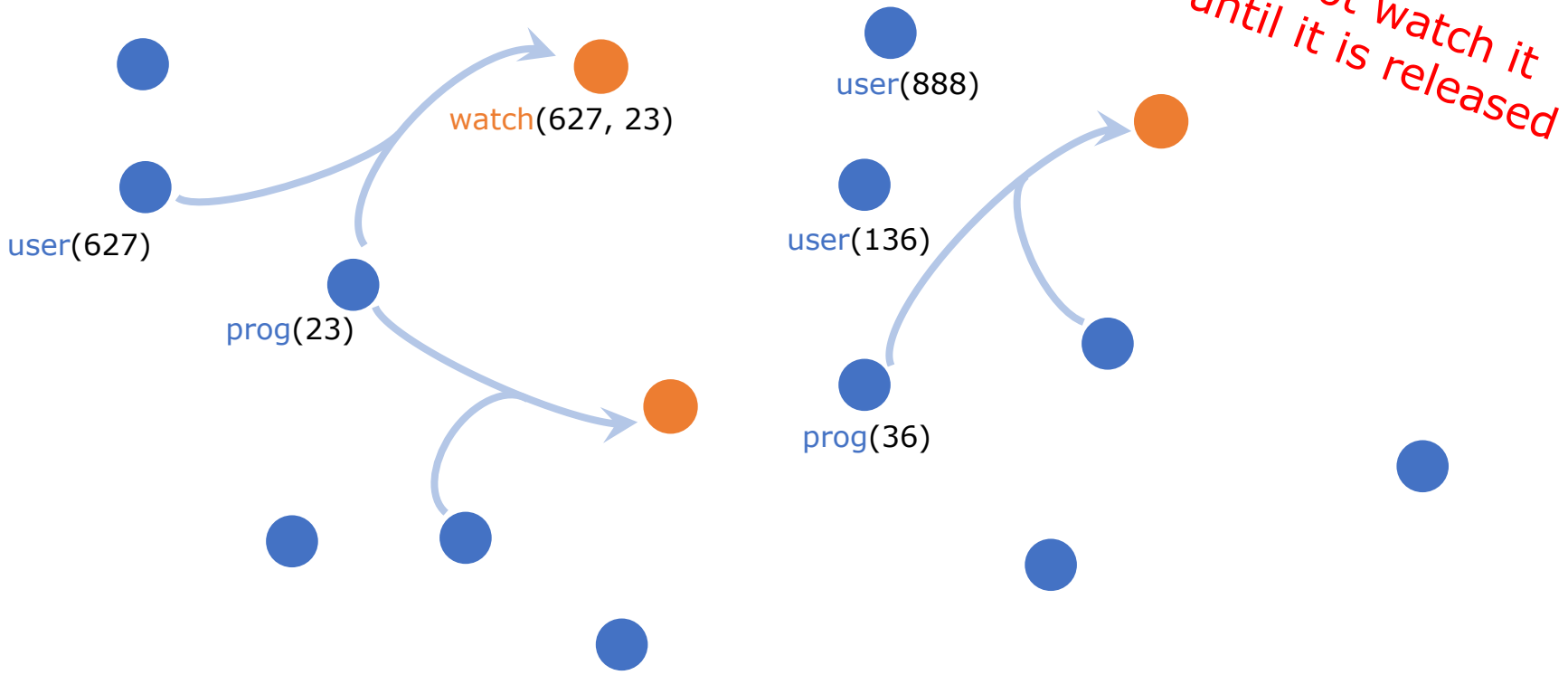
can not watch it
until it is released

Experiment: Users watch TV programs

collaborative filtering problem with timing
who watches what and when?

1000 users 49 TV programs to be released

49000 possible watch events

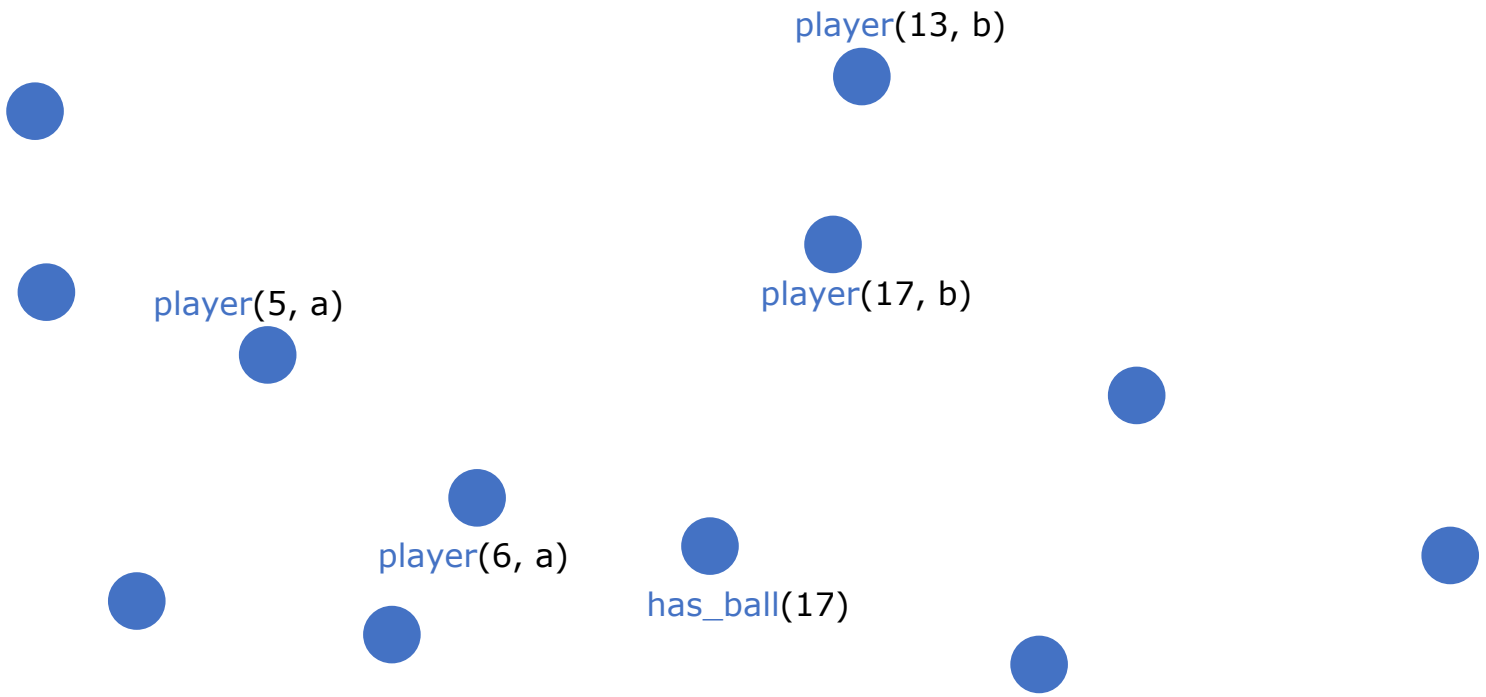


Experiment: Robots Kick/Pass/Steal

Experiment: Robots Kick/Pass/Steal

22 robot soccer players

player(Number, Team)

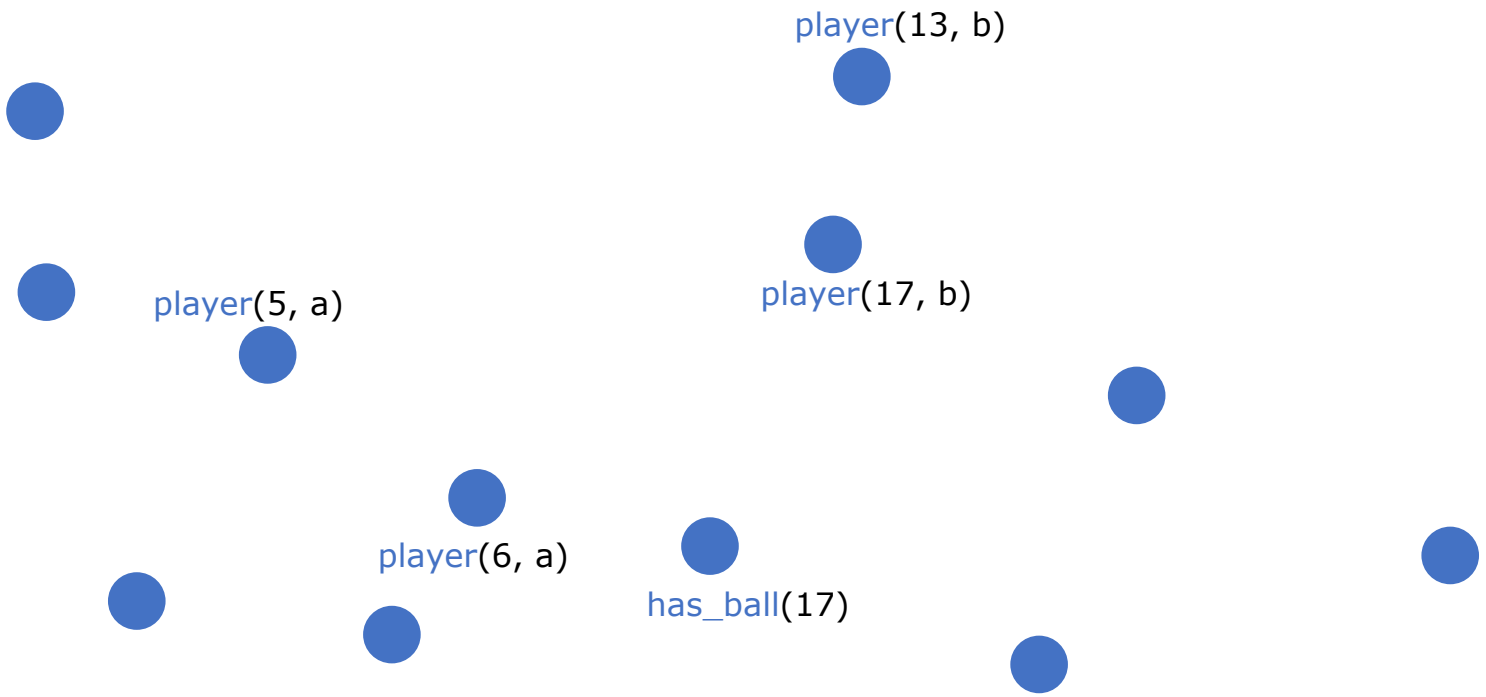


Experiment: Robots Kick/Pass/Steal

kick

22 robot soccer players

player(Number, Team)

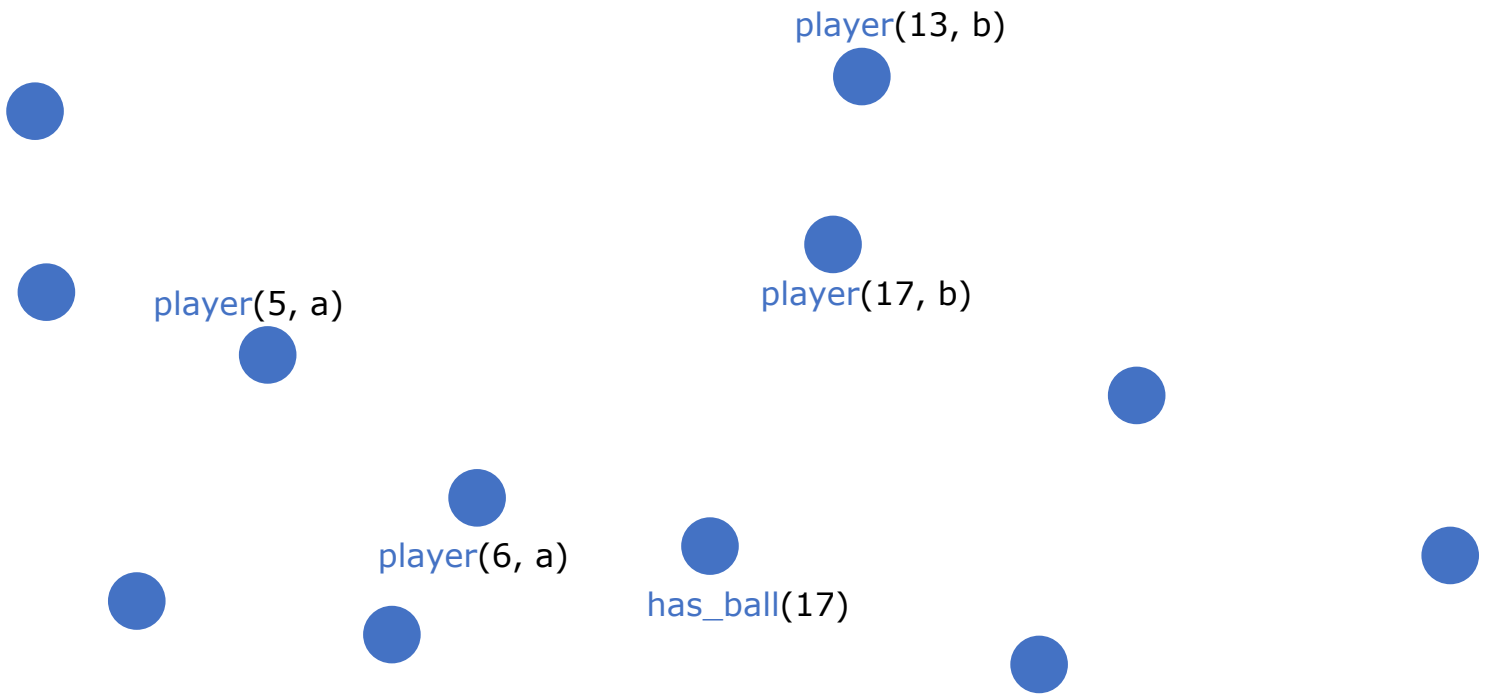


Experiment: Robots Kick/Pass/Steal

kick only if has ball

22 robot soccer players

player(Number, Team)



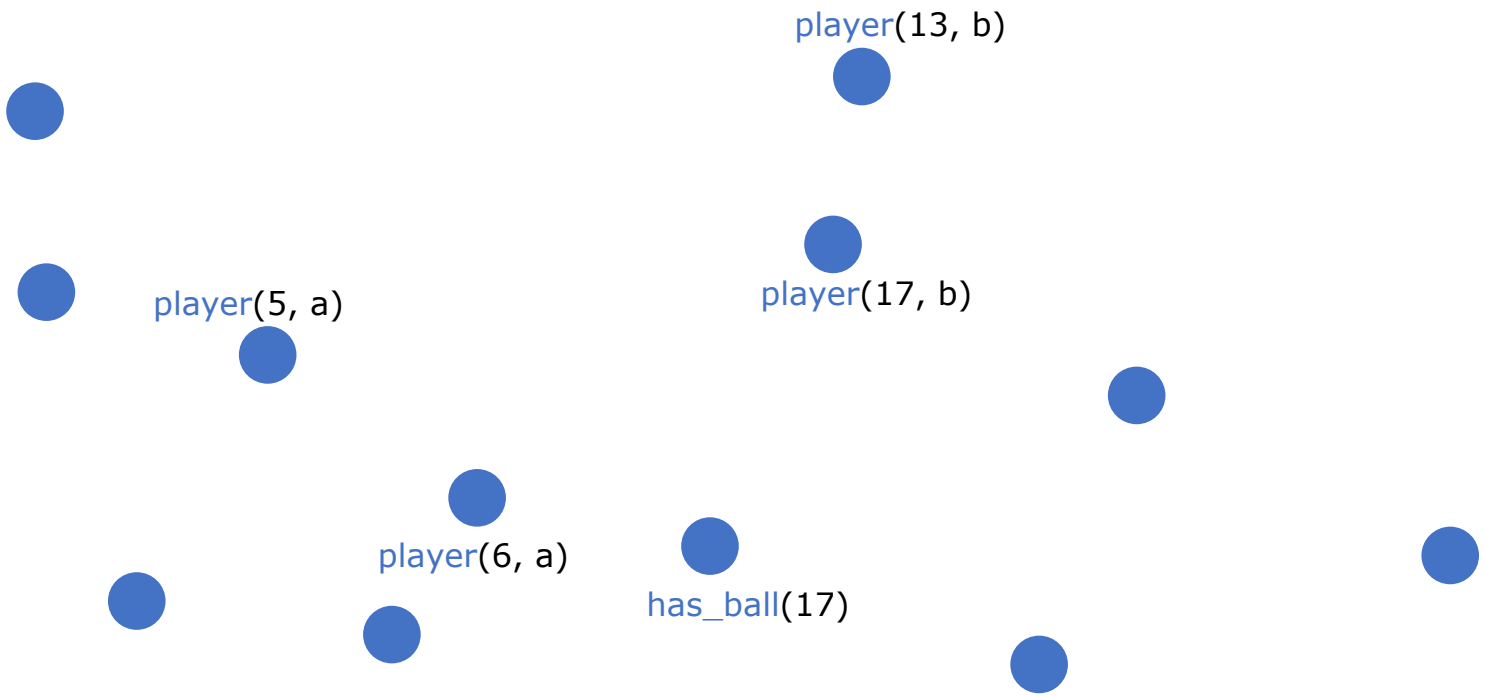
Experiment: Robots Kick/Pass/Steal

22 robot soccer players

player(Number, Team)

kick only if has ball

pass



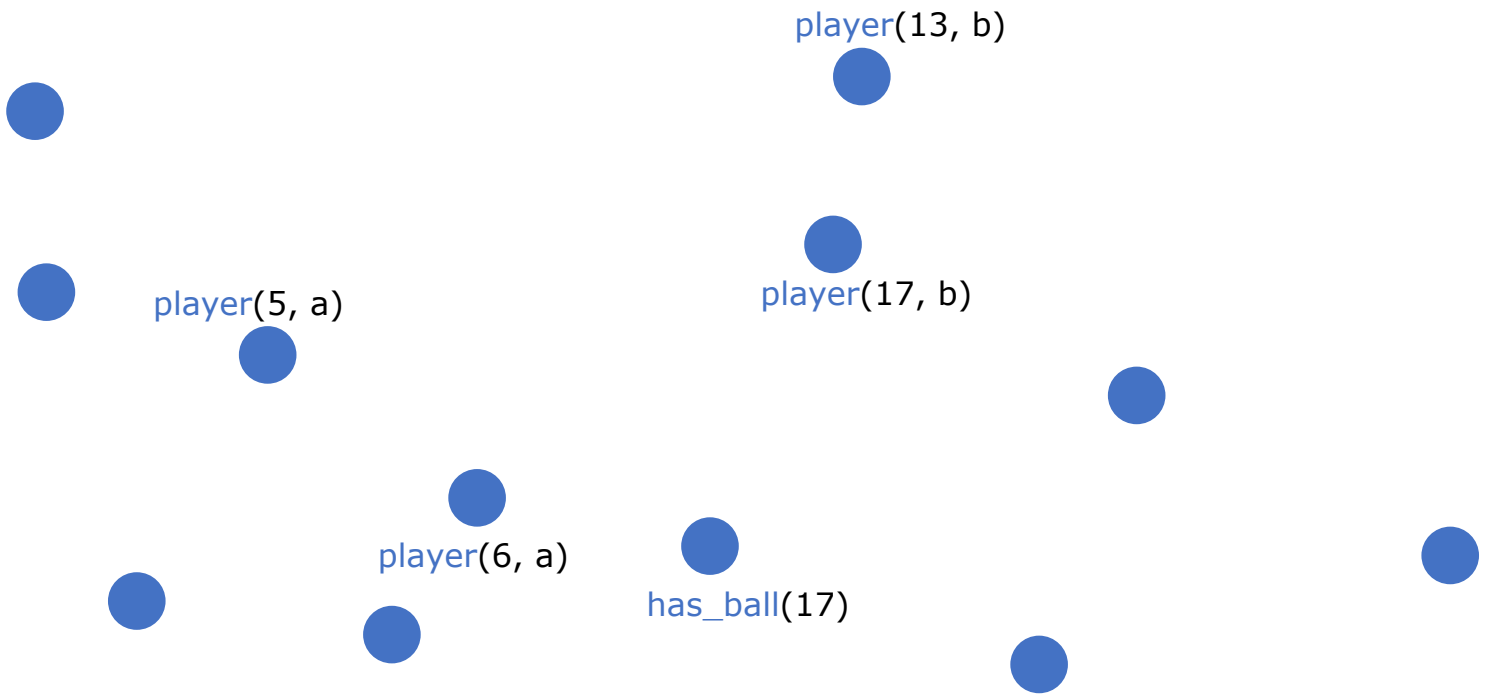
Experiment: Robots Kick/Pass/Steal

22 robot soccer players

player(Number, Team)

kick only if has ball

pass only to a teammate



Experiment: Robots Kick/Pass/Steal

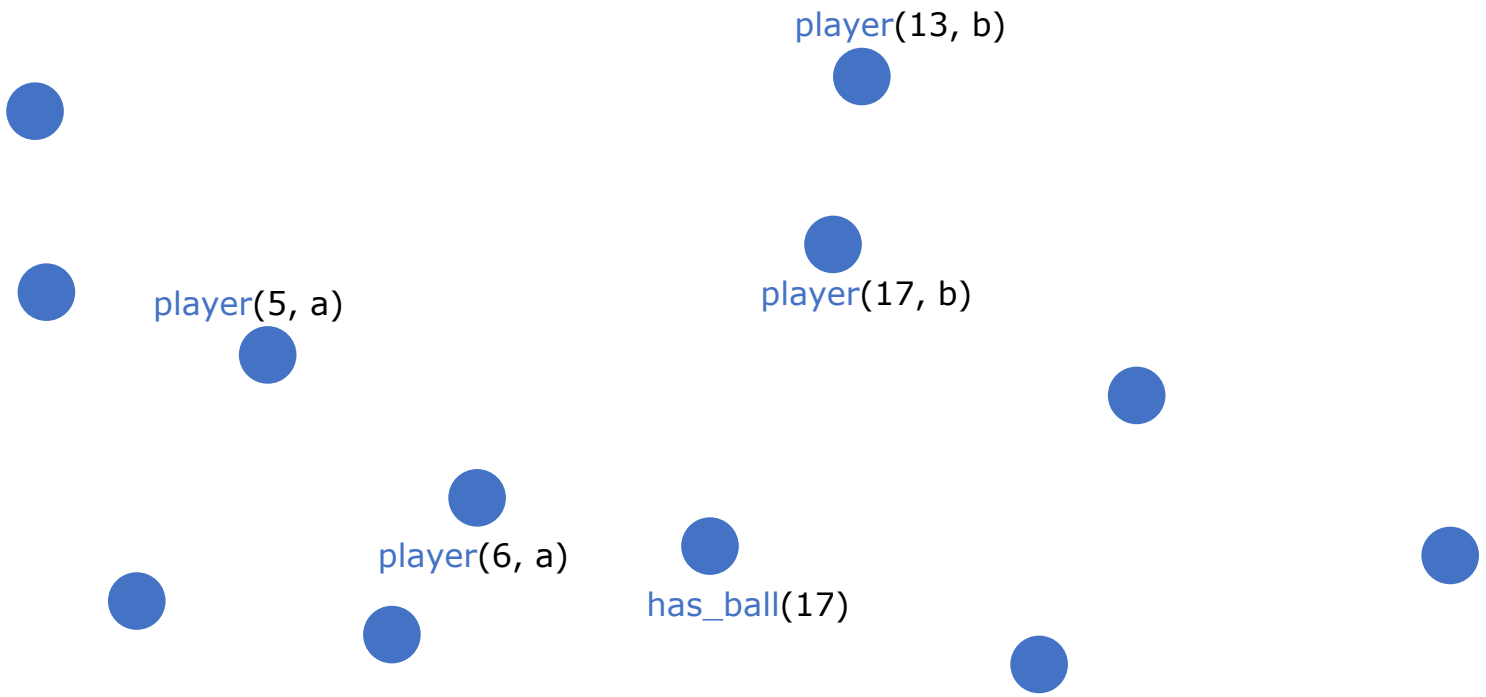
22 robot soccer players

player(Number, Team)

kick only if has ball

pass only to a teammate

steal

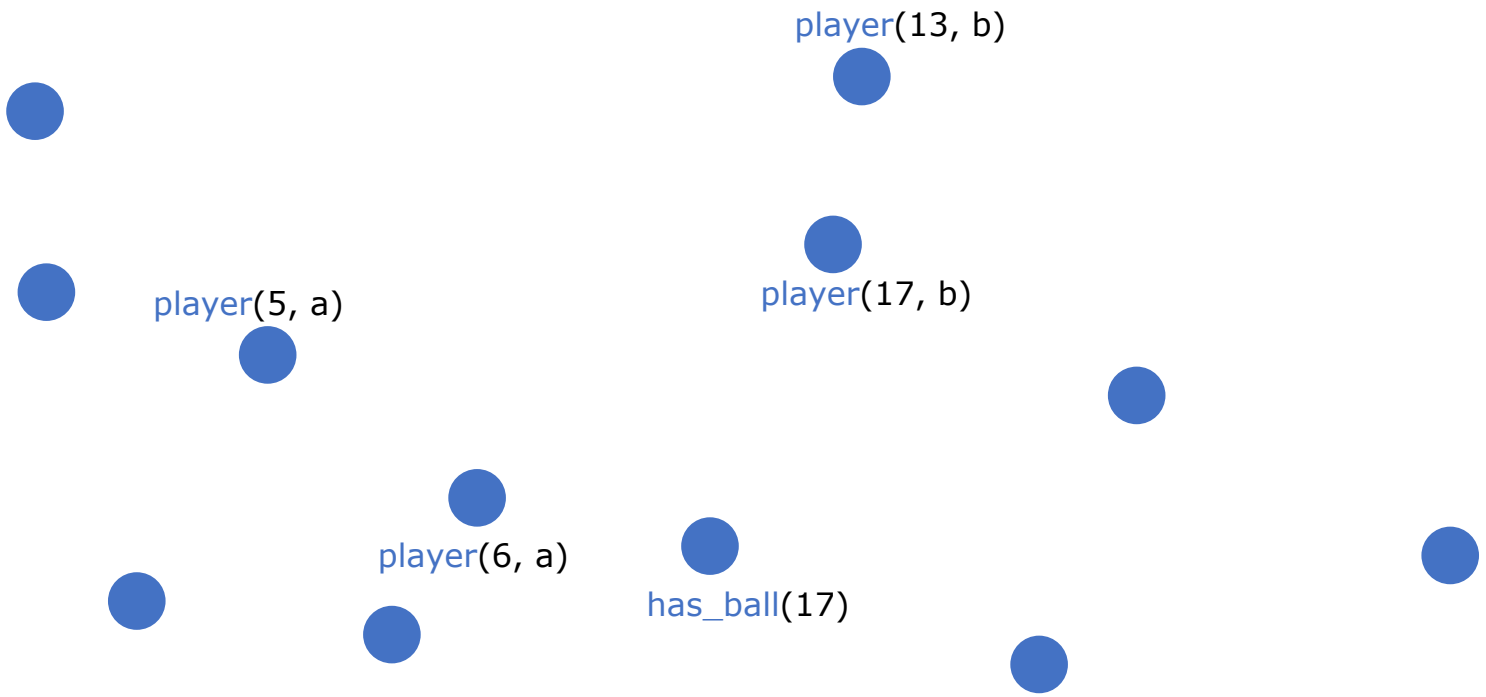


Experiment: Robots Kick/Pass/Steal

22 robot soccer players

player(Number, Team)

kick	only if has ball
pass	only to a teammate
steal	only from opponent

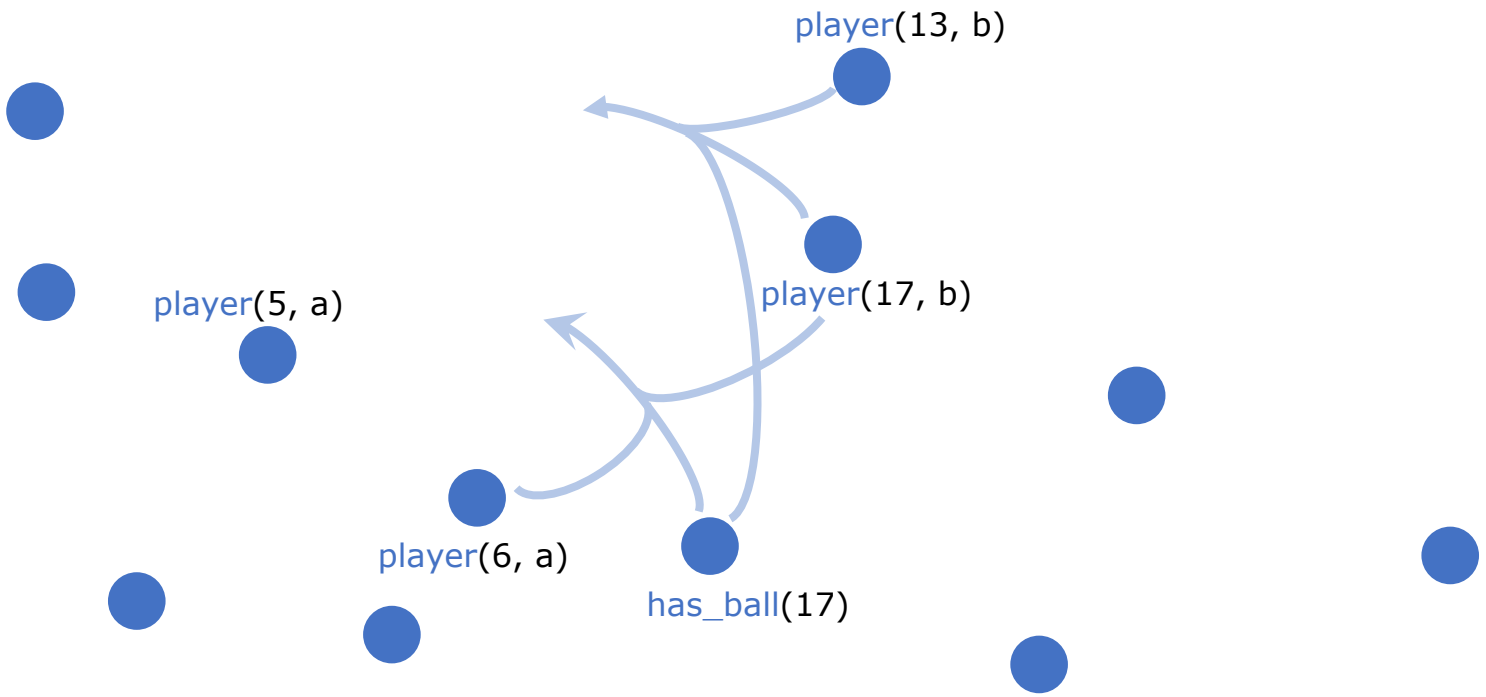


Experiment: Robots Kick/Pass/Steal

22 robot soccer players

player(Number, Team)

kick	only if has ball
pass	only to a teammate
steal	only from opponent

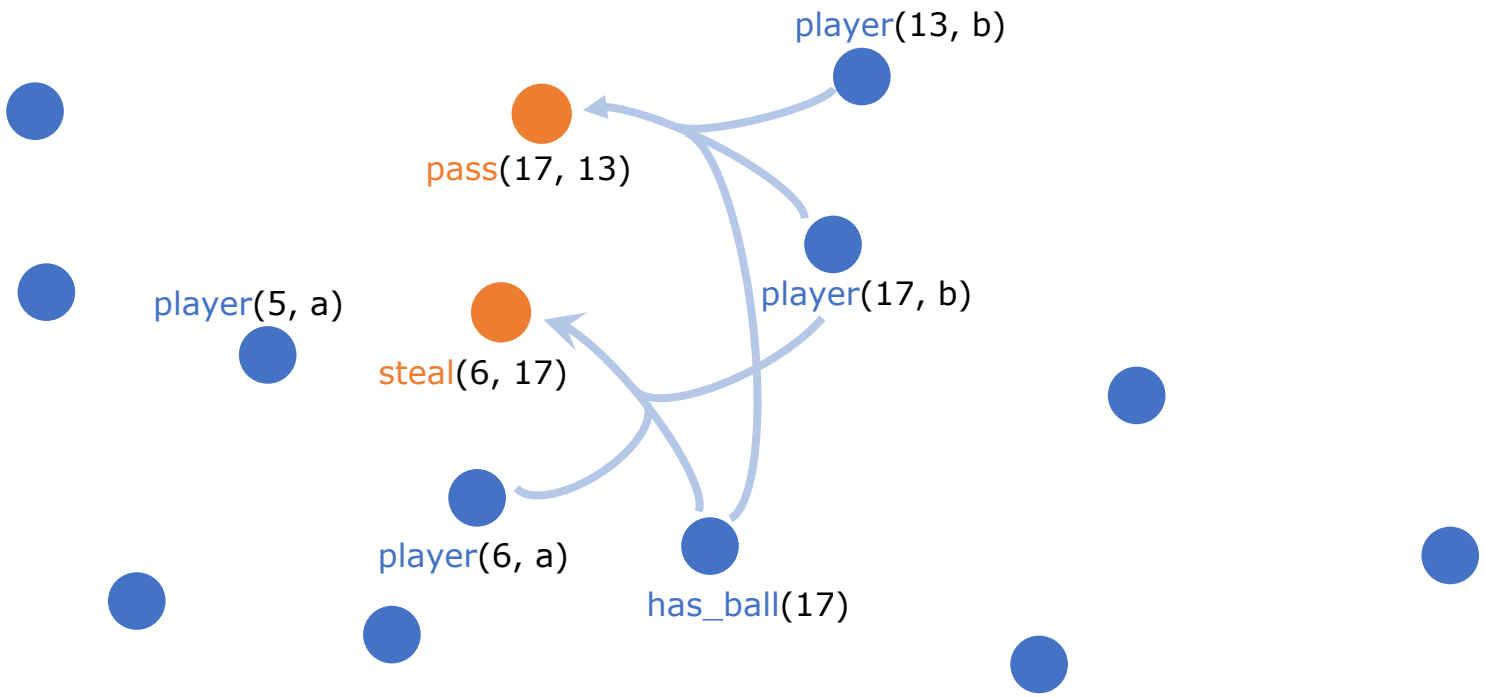


Experiment: Robots Kick/Pass/Steal

22 robot soccer players

player(Number, Team)

kick	only if has ball
pass	only to a teammate
steal	only from opponent

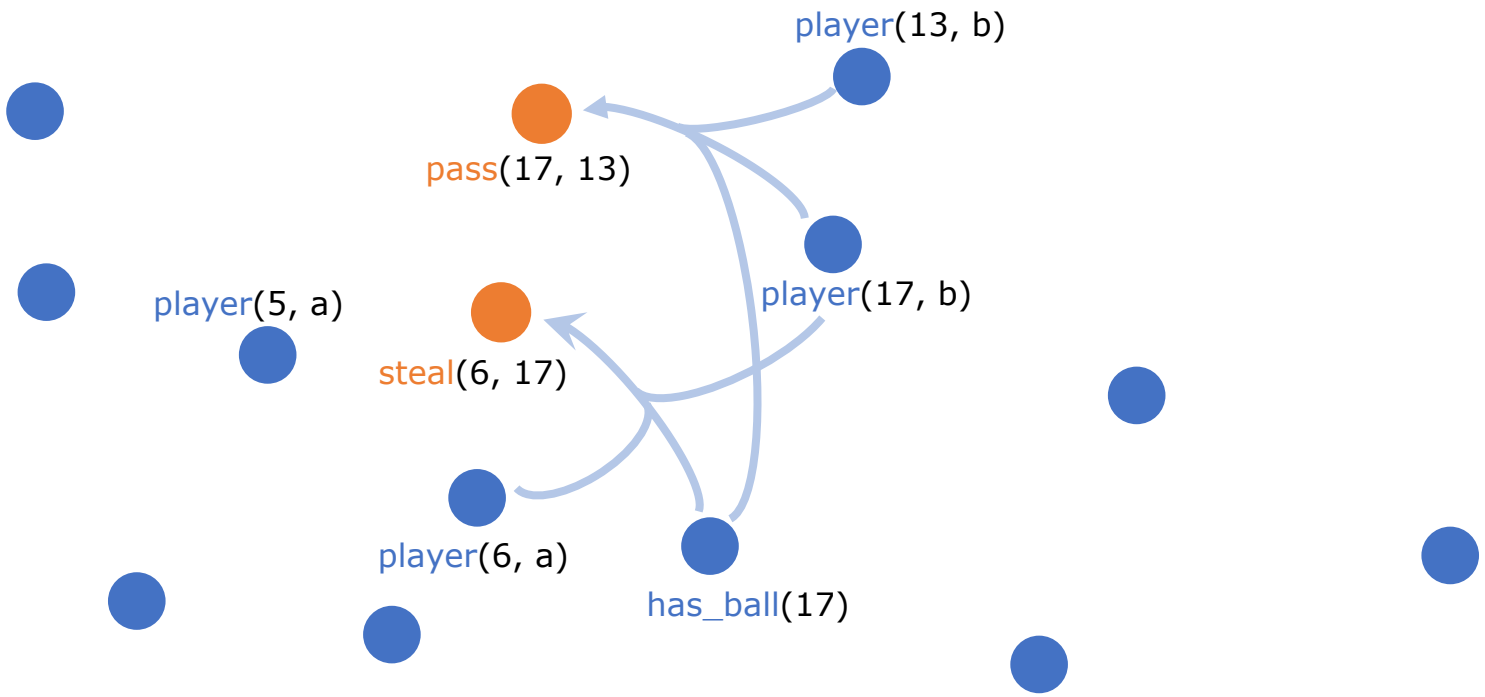


Experiment: Robots Kick/Pass/Steal

22 robot soccer players

player(Number, Team)

kick	only if has ball
pass	only to a teammate
steal	only from opponent

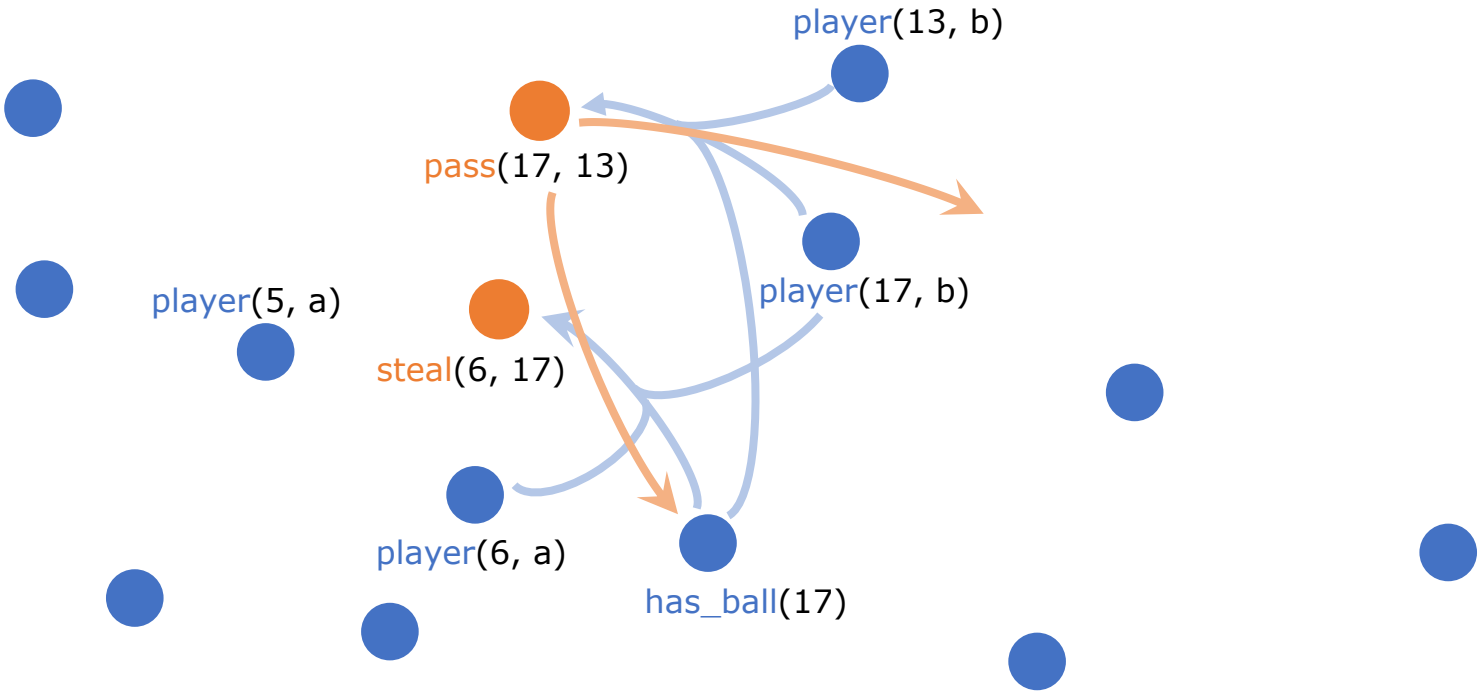


Experiment: Robots Kick/Pass/Steal

22 robot soccer players

player(Number, Team)

kick	only if has ball
pass	only to a teammate
steal	only from opponent

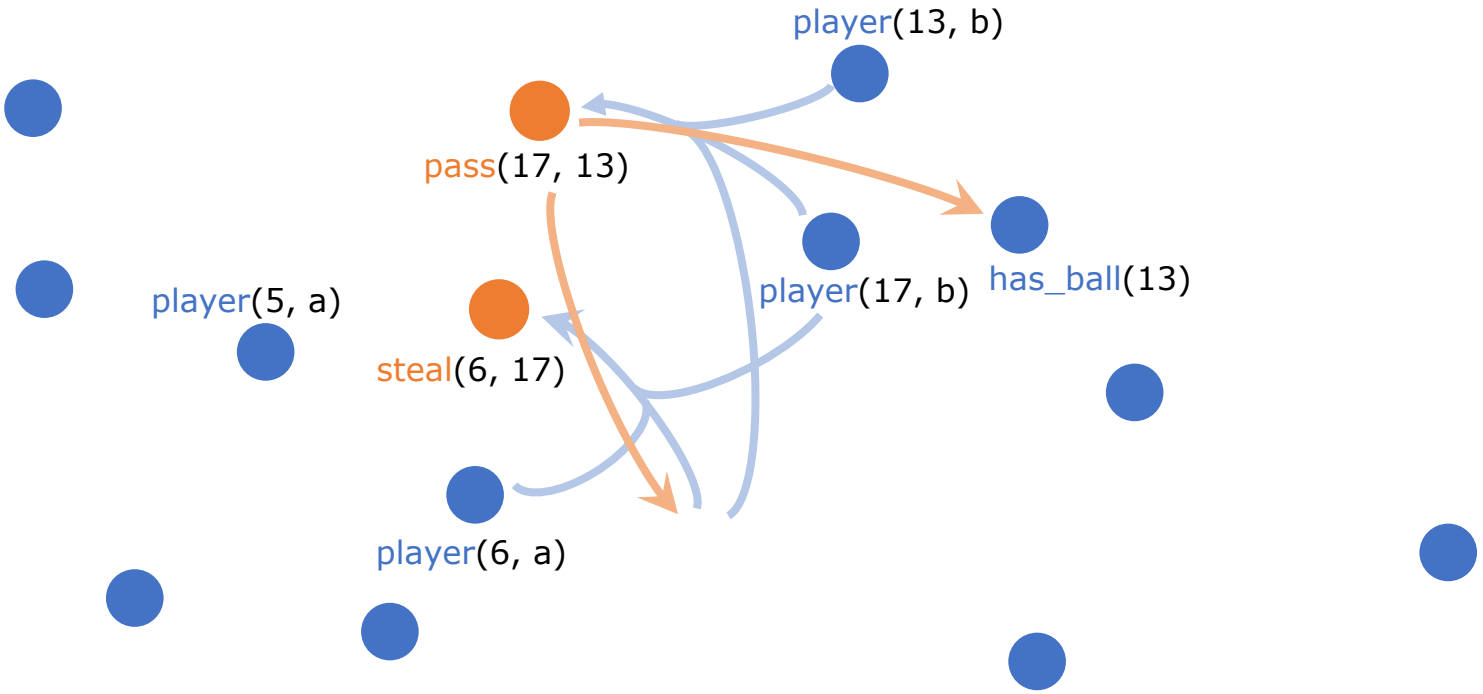


Experiment: Robots Kick/Pass/Steal

22 robot soccer players

player(Number, Team)

kick	only if has ball
pass	only to a teammate
steal	only from opponent

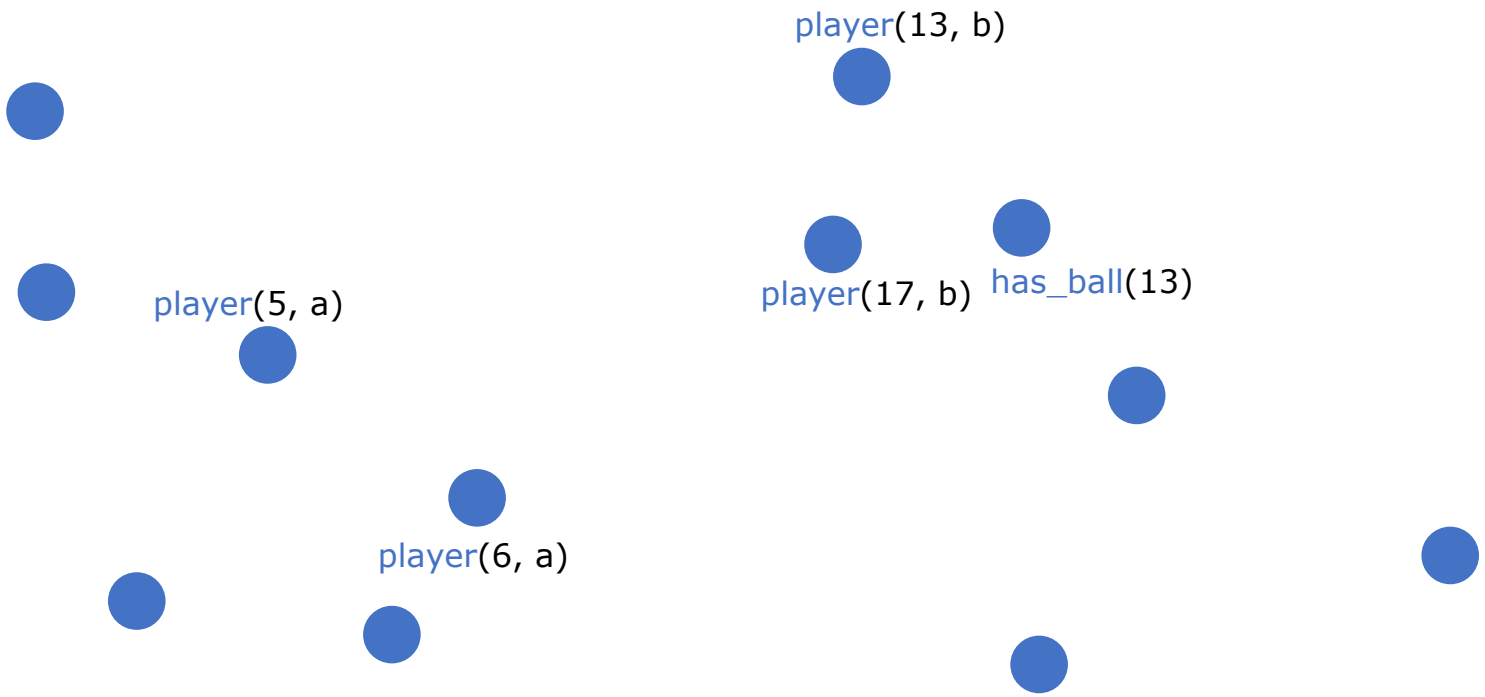


Experiment: Robots Kick/Pass/Steal

22 robot soccer players

player(Number, Team)

kick	only if has ball
pass	only to a teammate
steal	only from opponent



Experiment: Robots Kick/Pass/Steal

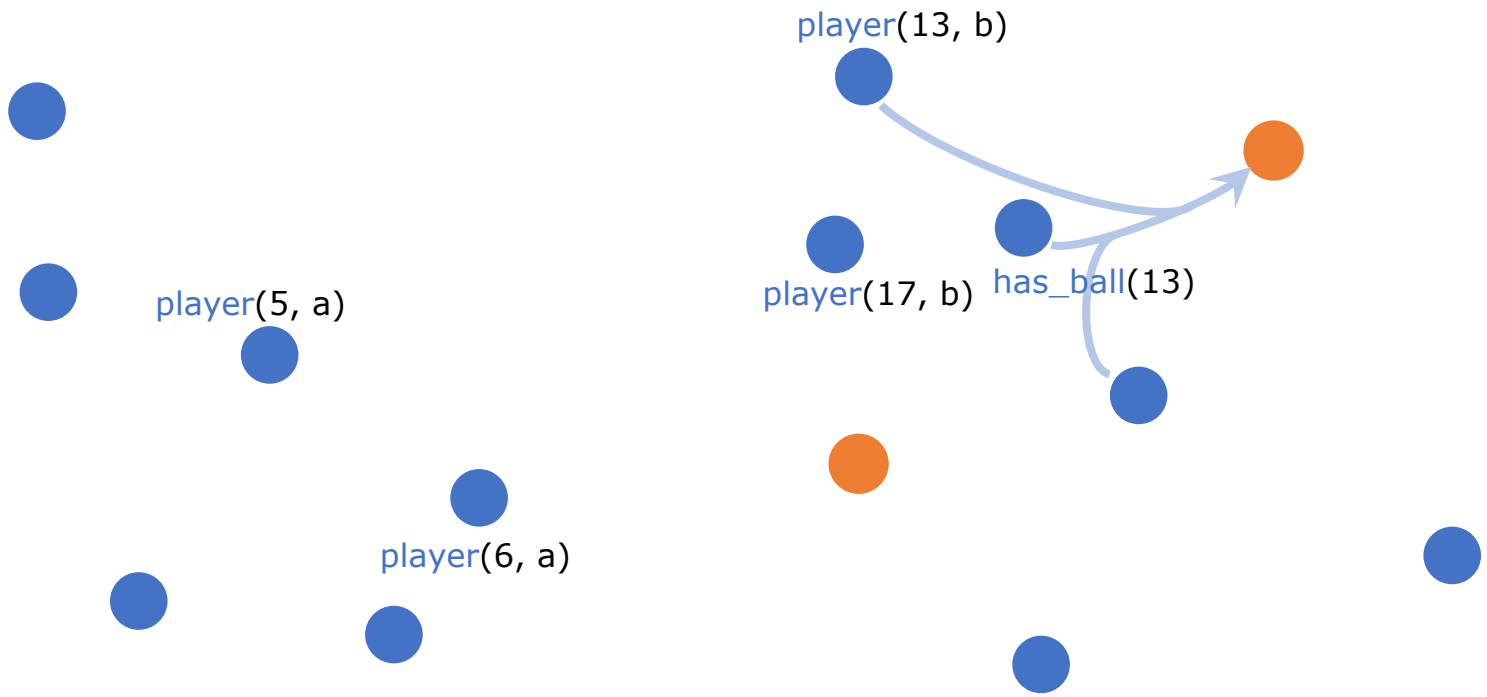
22 robot soccer players

player(Number, Team)

kick only if has ball

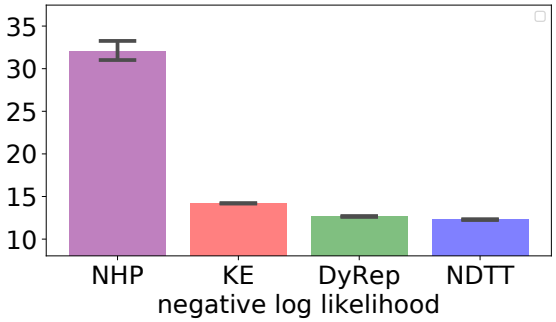
pass only to a teammate

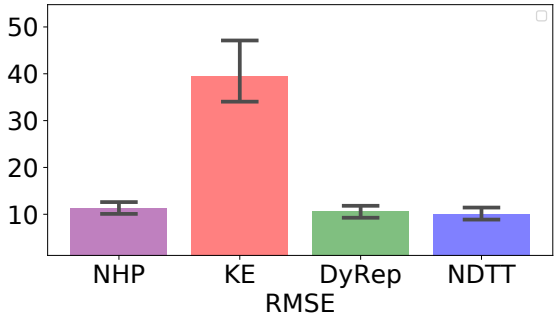
steal only from opponent

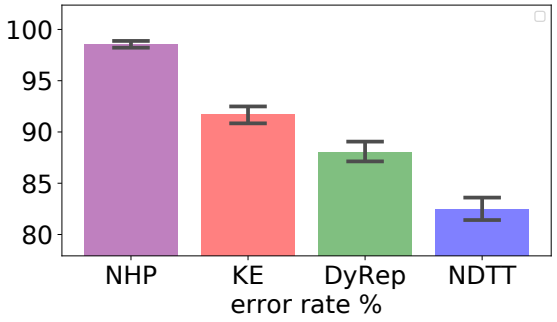


Results: NDTT > Competitors

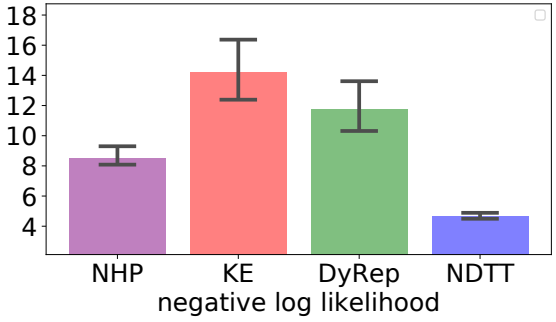
3 error metrics (in 3 columns): smaller is better

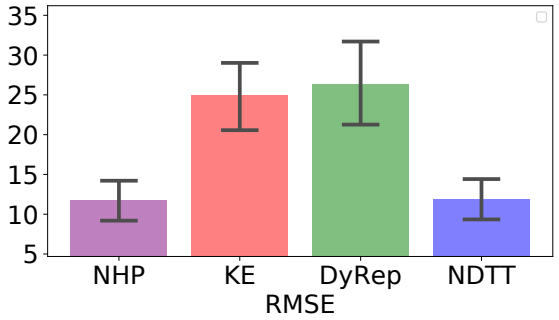


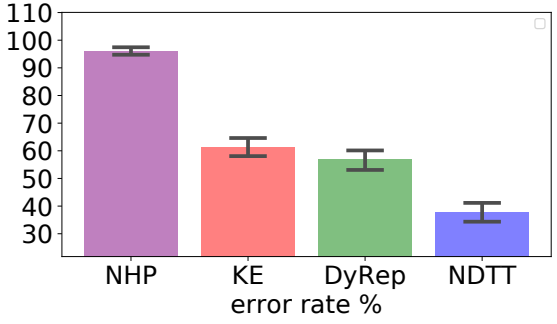




users watch TV programs

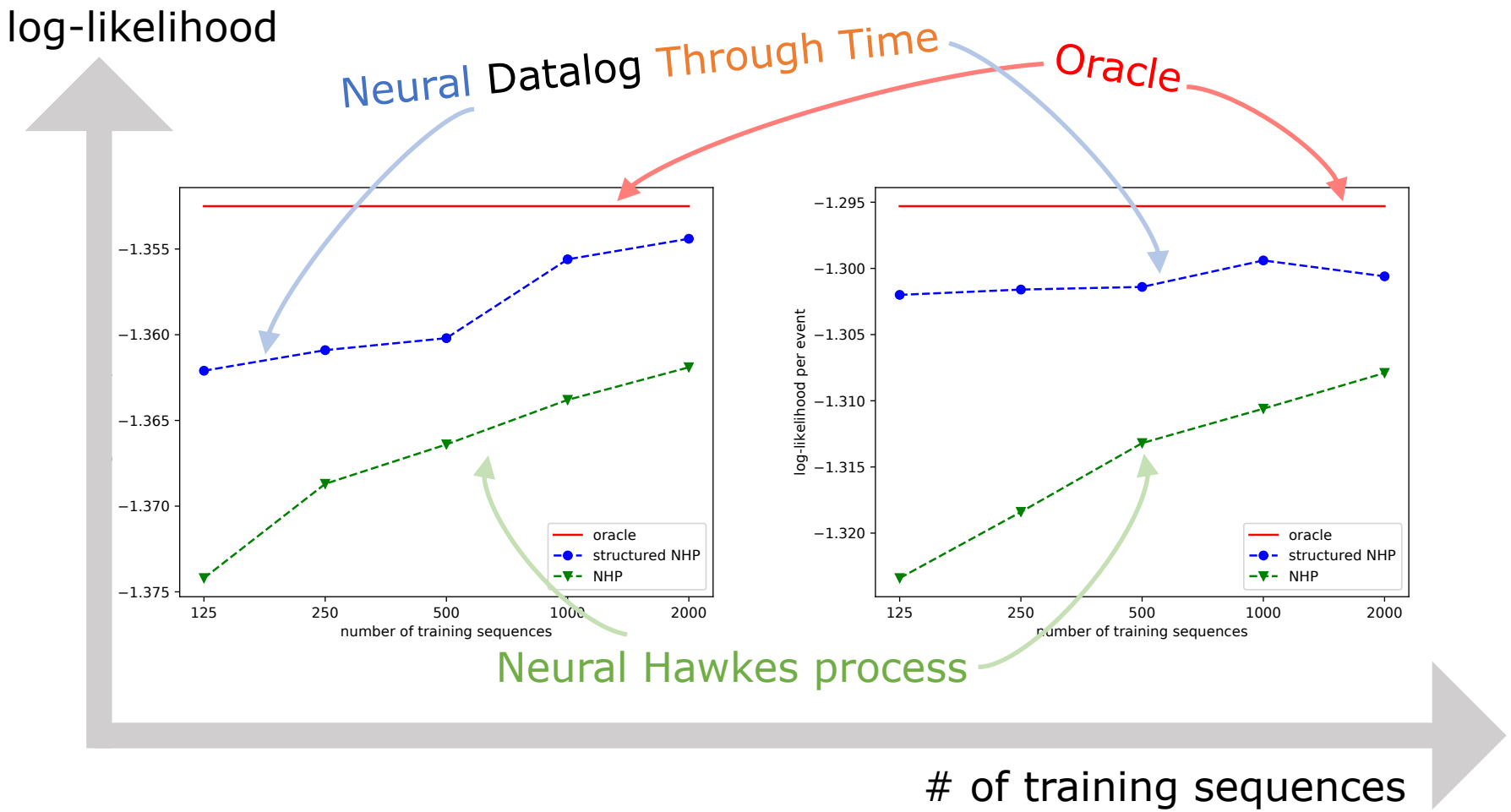






robots kick/pass/steal soccer ball

Good Generalization with Less Data



Summary: Deep Recurrent Net

Summary: **Deep Recurrent Net**

e.g., RNN
LSTM

discrete-time

Summary: **Deep Recurrent Net**

e.g., RNN discrete-time
LSTM
neural Hawkes process continuous-time

Summary: **Deep Recurrent Net**

hidden
system
state

e.g., RNN discrete-time
LSTM
neural Hawkes process continuous-time

Summary: **Deep Recurrent Net**

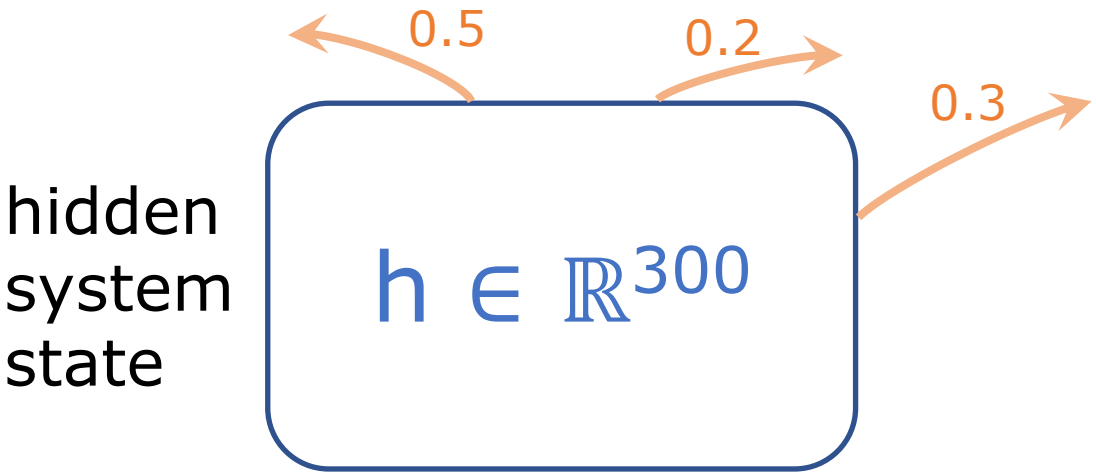
hidden system state

e.g., RNN discrete-time

LSTM

neural Hawkes process continuous-time

Summary: Deep Recurrent Net

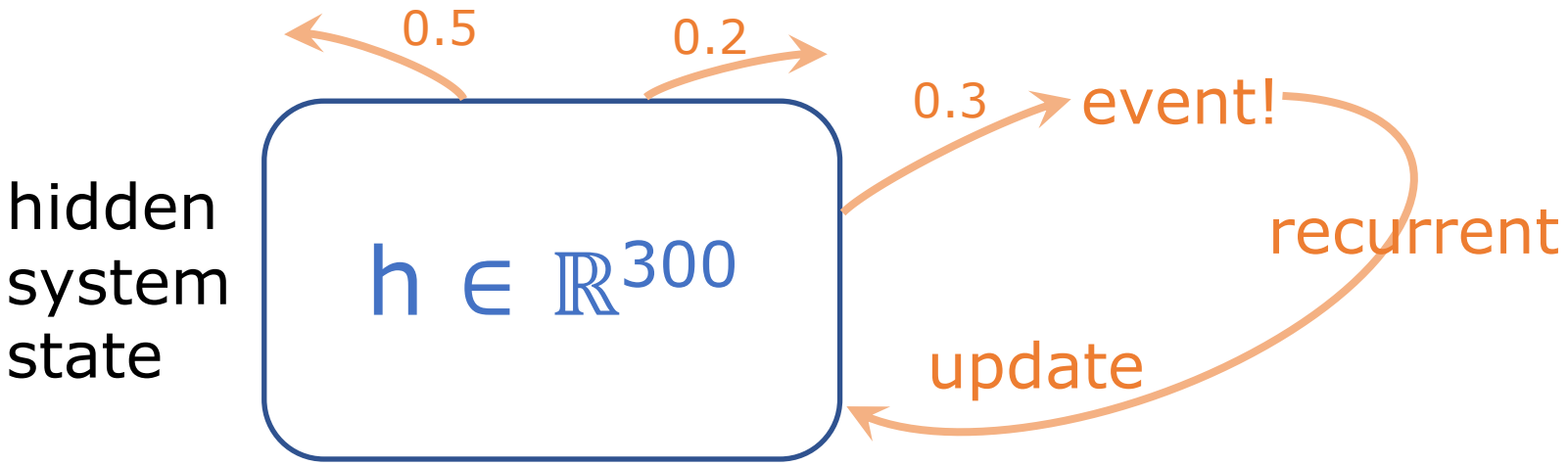


e.g., RNN discrete-time
LSTM
neural Hawkes process continuous-time

Summary: Deep Recurrent Net

e.g., RNN discrete-time
LSTM
neural Hawkes process continuous-time

Summary: Deep Recurrent Net



e.g., RNN discrete-time
LSTM
neural Hawkes process continuous-time

Summary: Logic ➔ Deep Recurrent Net

hidden system state

Summary: Logic ➔ Deep Recurrent Net

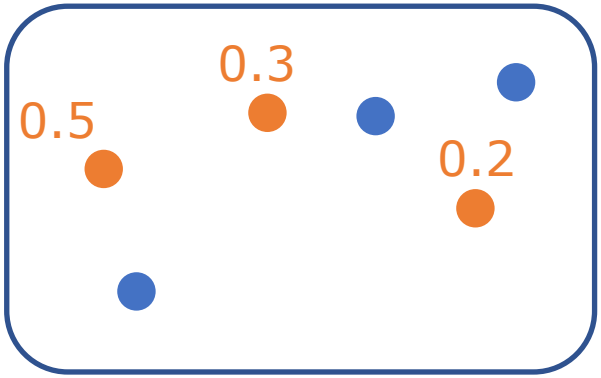
distributed

hidden
system
state

Summary: Logic ➔ Deep Recurrent Net

distributed

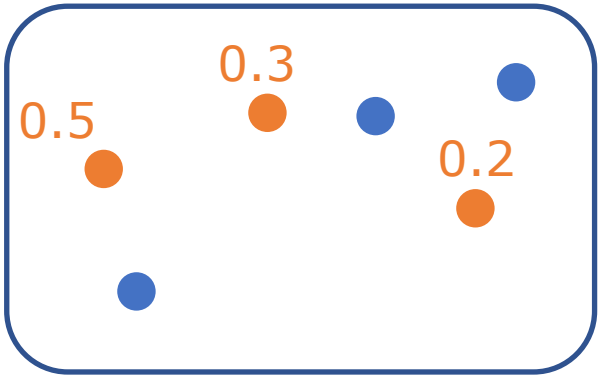
hidden
system
state



Summary: Logic ➔ Deep Recurrent Net

distributed

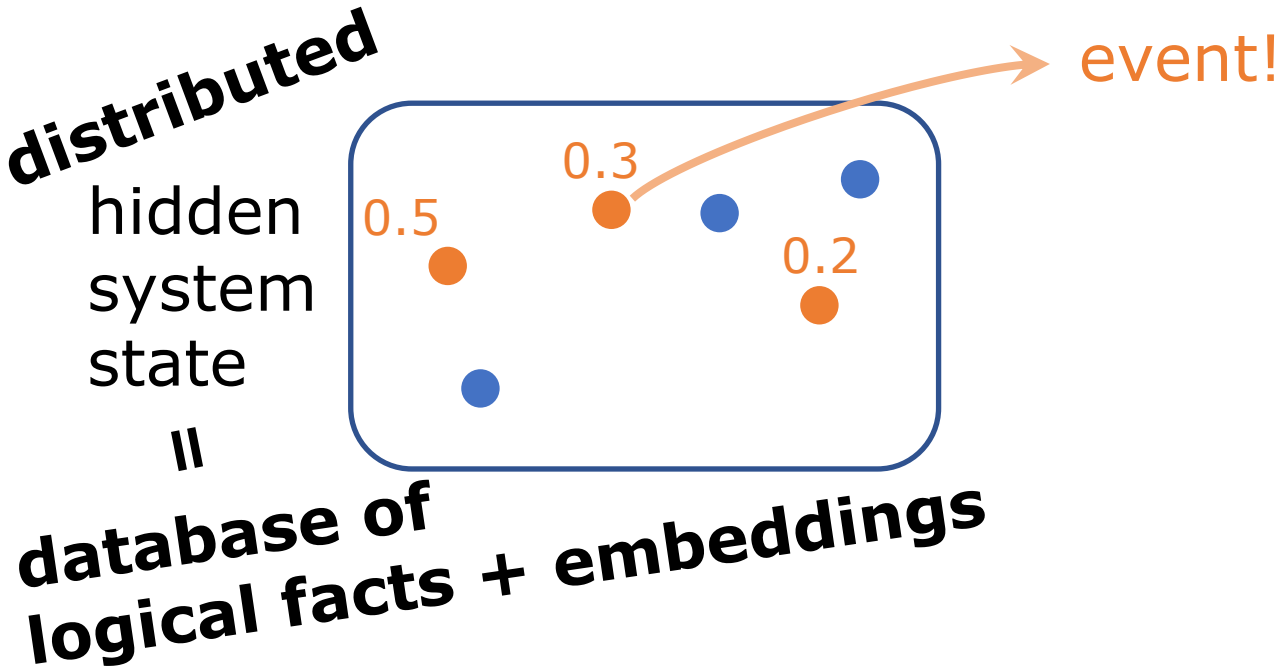
hidden
system
state



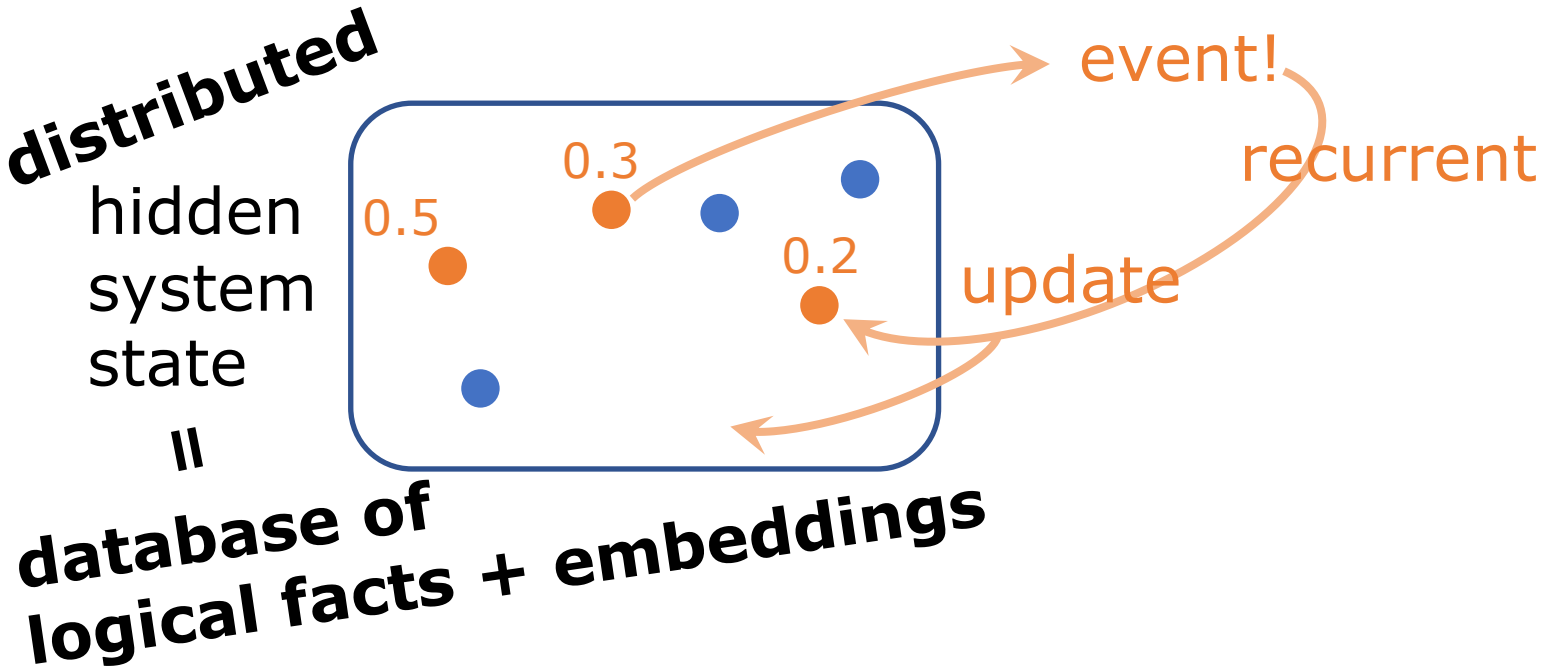
II

database of
logical facts + embeddings

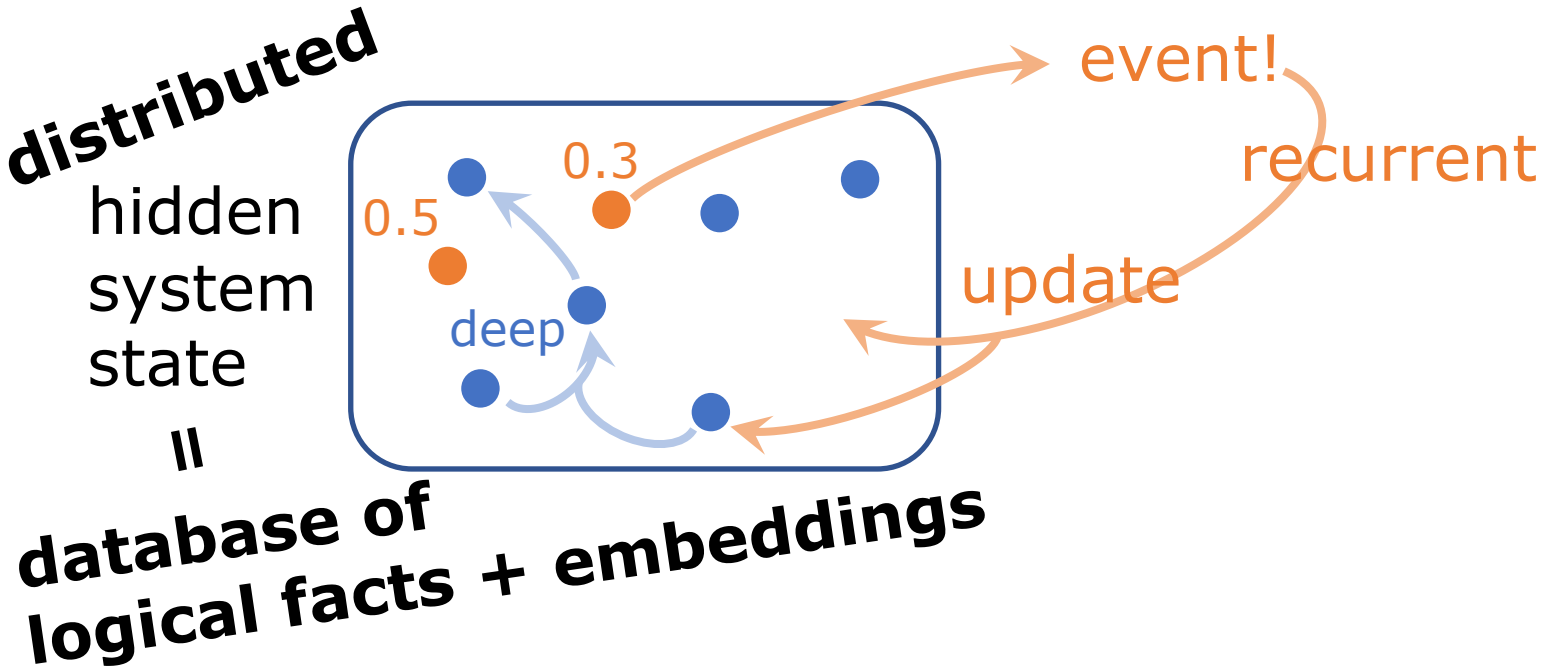
Summary: Logic ➔ Deep Recurrent Net



Summary: Logic ➔ Deep Recurrent Net



Summary: Logic ➔ Deep Recurrent Net



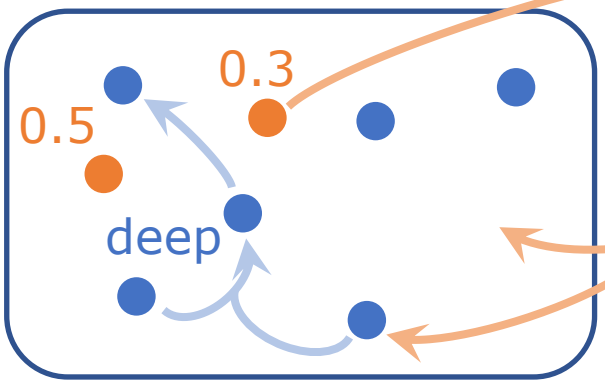
Summary: Logic → Deep Recurrent Net

distributed

hidden
system
state

=

database of logical facts + embeddings



event!

recurrent

update

update derived from small rule set + learned low-dim matrices

new fact :- old fact $_1$, …

new fact ← event, …

! old fact ← event, …

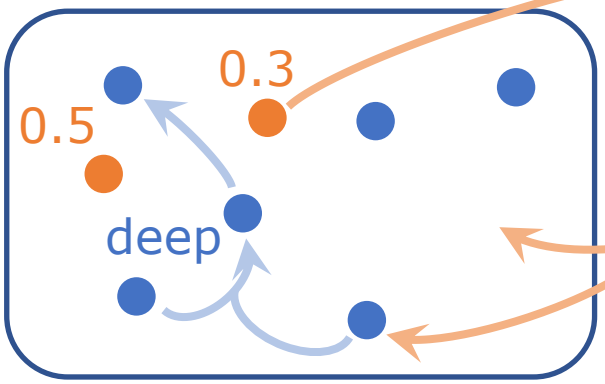
Summary: Logic → Deep Recurrent Net

distributed

hidden
system
state

=

database of logical facts + embeddings



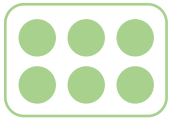
event!

recurrent

update

update derived from small rule set + learned low-dim matrices

new fact :- old fact $_1$, …



new fact ← event, …

! old fact ← event, …

try our code in your domain!

Neural Datalog Through Time