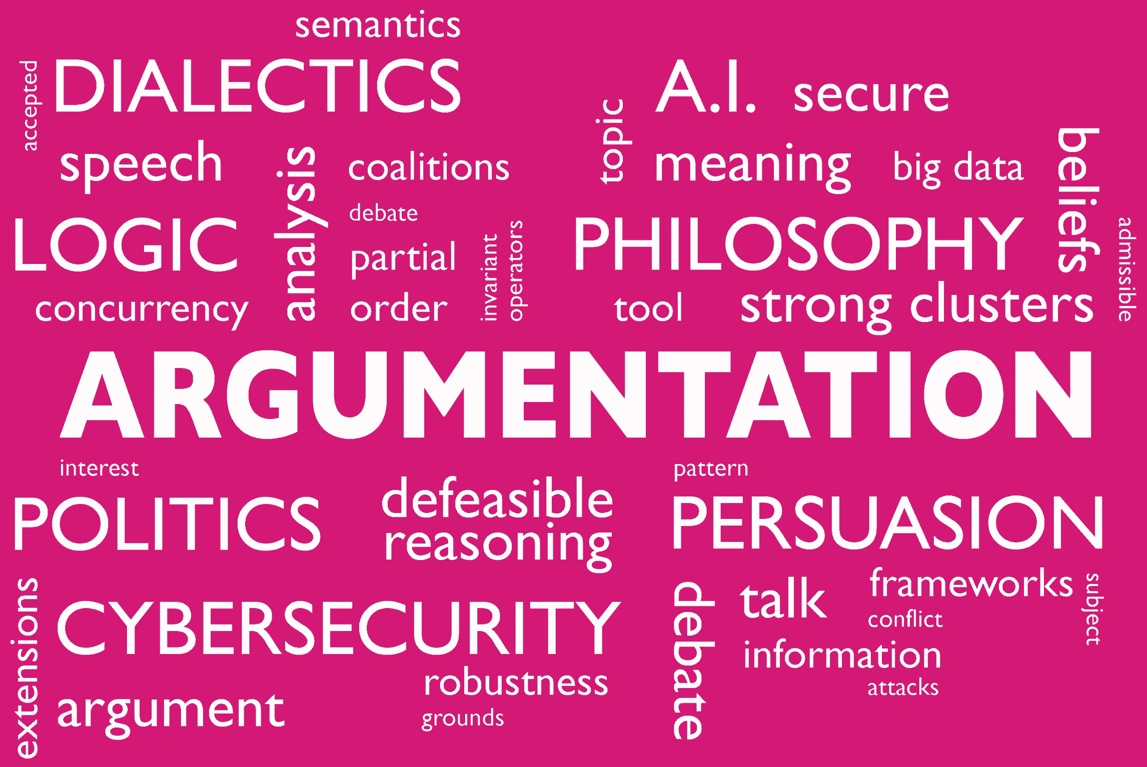
UNIVERSITÀ DEGLI STUDI
DI PERUGIA

Università degli Studi "G. d'Annunzio"

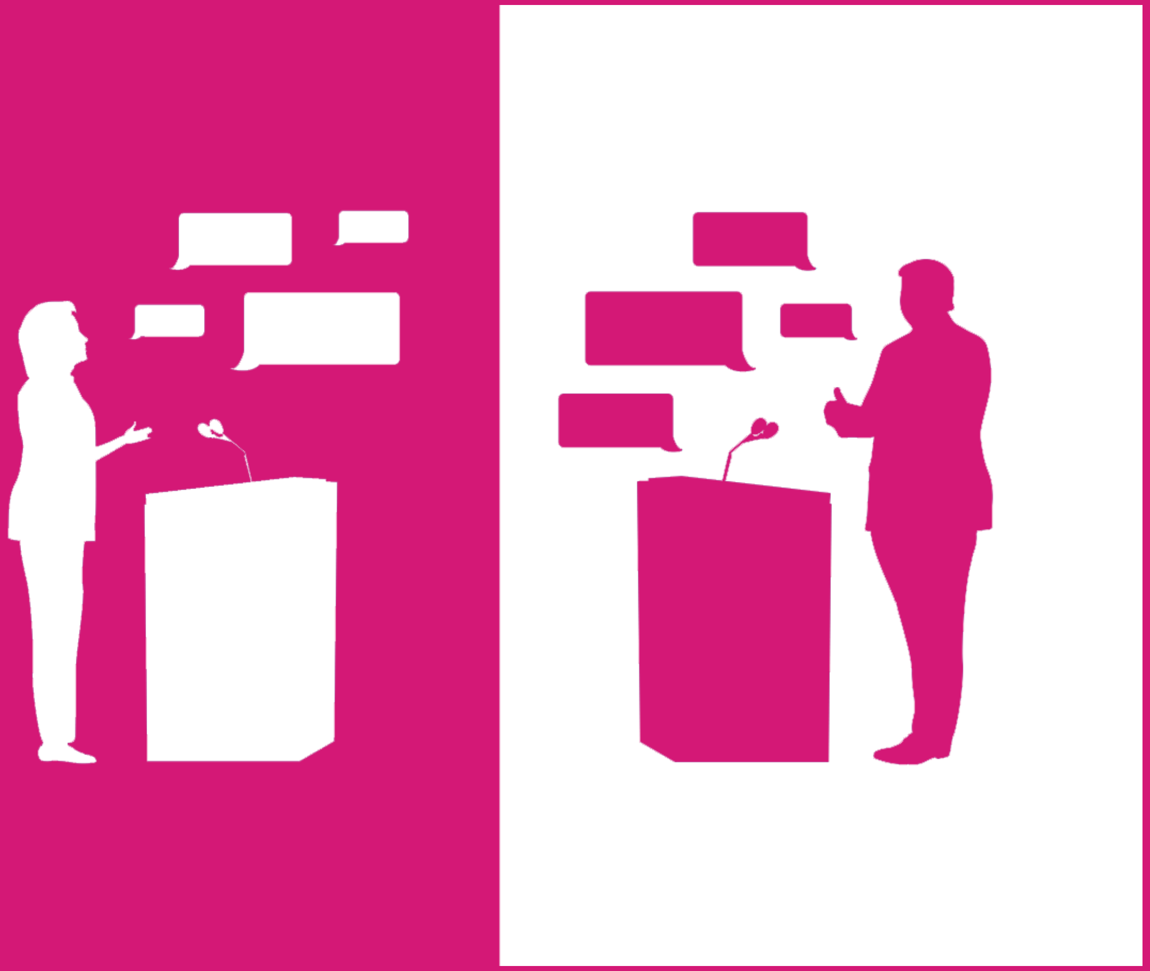
Timed Concurrent Language for Argumentation

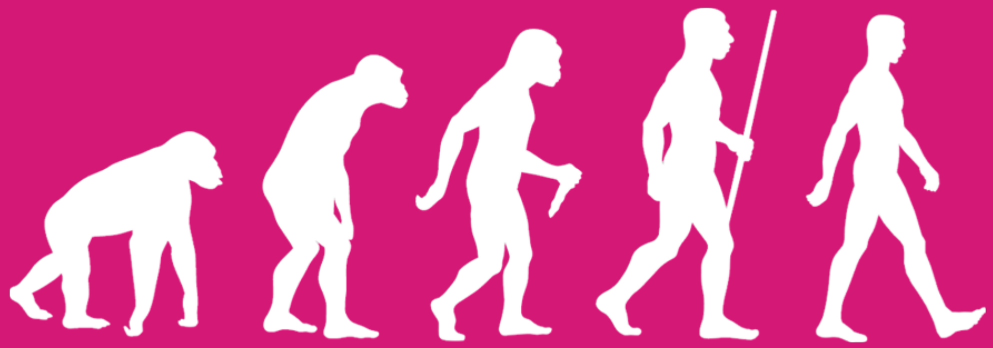
Stefano Bistarelli, Maria Chiara Meo, <u>Carlo Taticchi</u>

CILC 2021


semantics
accepted
DIALECTICS
A.I.
secure
speech
analysis
coalitions
topic
meaning
big data
beliefs
debate
LOGIC
partial
order
invariant
operators
PHILOSOPHY
admissible
concurrency
tool
strong clusters
ARGUMENTATION
interest
pattern
POLITICS
defeasible
reasoning
PERSUASION
CYBERSECURITY
debate
talk
frameworks
subject
conflict
information
attacks
extensions
robustness
argument
grounds

30

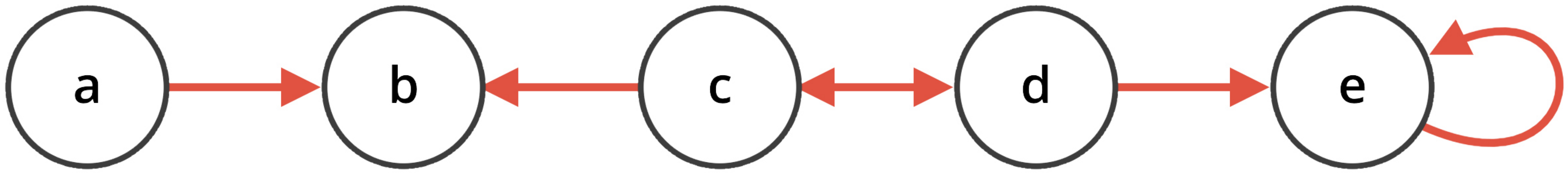




Overview

- Argumentation Frameworks (AFs)
  - Acceptable arguments
  - Timed AFs
- Timed Concurrent Language for Argumentation (`tcla`)
  - Syntax
  - Operational semantics
- Modelling Timed AFs
  - Running example
- Conclusion and Future Work

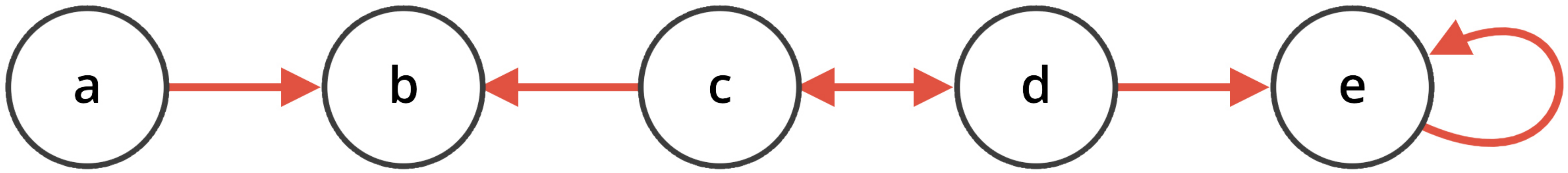
Argumentation Frameworks



Argumentation Frameworks

Acceptable arguments

- Conflict-Free: `{}, {a}, {b}, {c}, {d}, {a,c}, {a,d}, {b,d}`
- Admissible: `{}, {a}, {b}, {c}, {d}, {a,c}, {a,d}, {b,d}`
- Complete: `{}, {a}, {b}, {c}, {d}, {a,c}, {a,d}, {b,d}`
- …



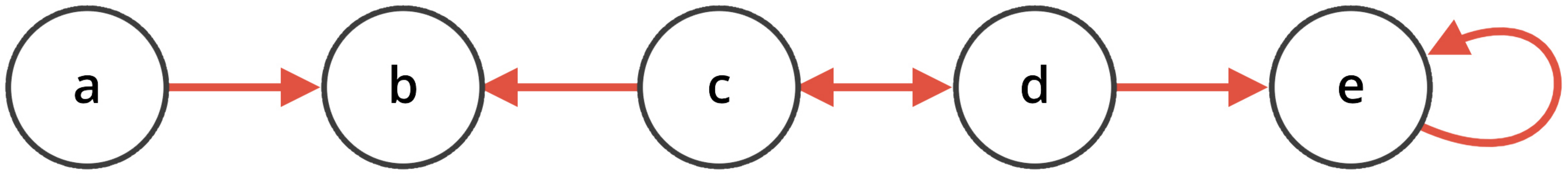
Extension-based semantics

Argumentation Frameworks

Acceptable arguments

- Conflict-Free: `{}, {a}, {b}, {c}, {d}, {a,c}, {a,d}, {b,d}`
- Admissible: `{}, {a}, {b}, {c}, {d}, {a,c}, {a,d}, {b,d}`
- Complete: `{}, {a}, {b}, {c}, {d}, {a,c}, {a,d}, {b,d}`
- …

Credulously accepted

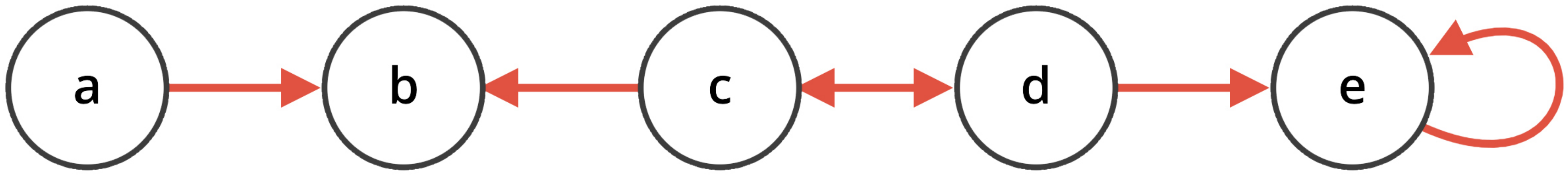


Extension-based semantics

Argumentation Frameworks

Acceptable arguments

- Conflict-Free: {}, {a}, {b}, {c}, {d}, {a,c}, {a,d}, {b,d}
- Admissible: {}, {a}, {b}, {c}, {d}, {a,c}, {a,d}, {b,d}
- Complete: {}, {a}, {b}, {c}, {d}, {a,c}, {a,d}, {b,d}
- … Skeptically accepted Credulously accepted



Extension-based semantics

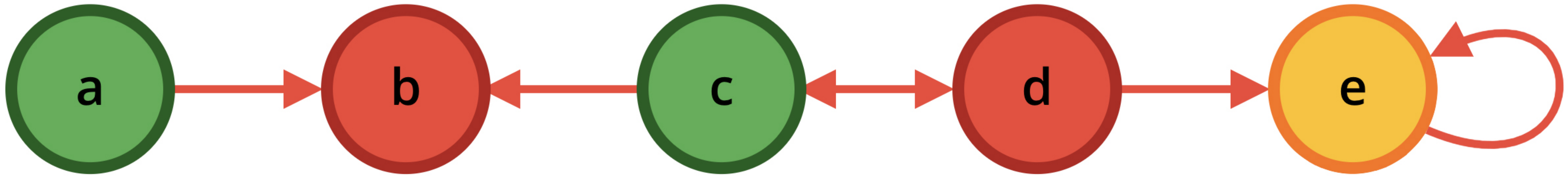
Argumentation Frameworks

Acceptable arguments

IN if it is attacked only by **OUT** arguments

OUT if it is attacked by at least an **IN** argument

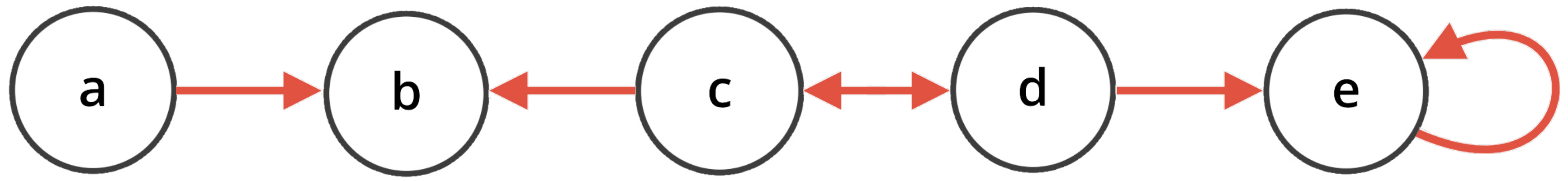
UNDEC otherwise

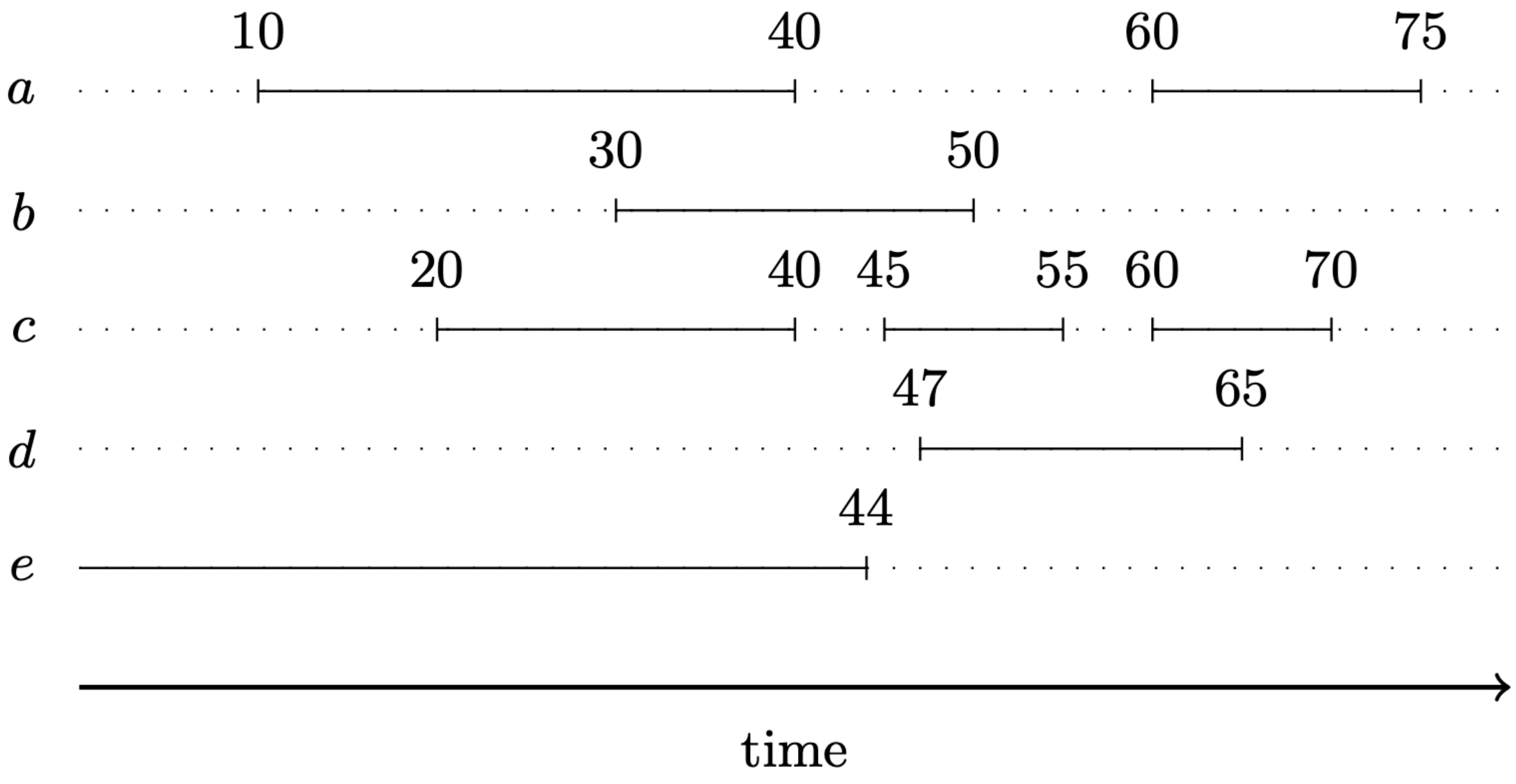


Reinstatement labelling identifies complete extensions

Argumentation Frameworks

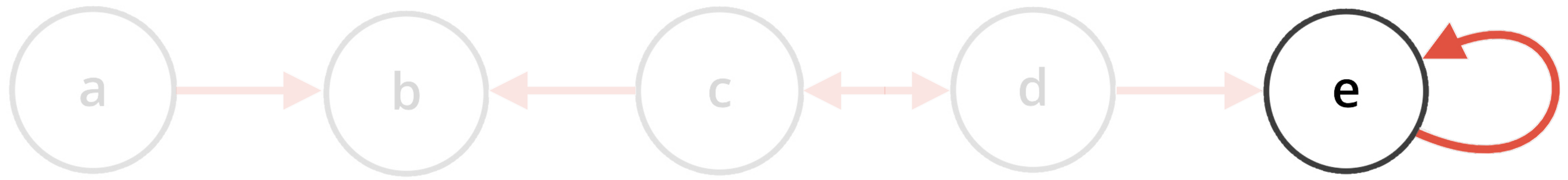
Timed AFs

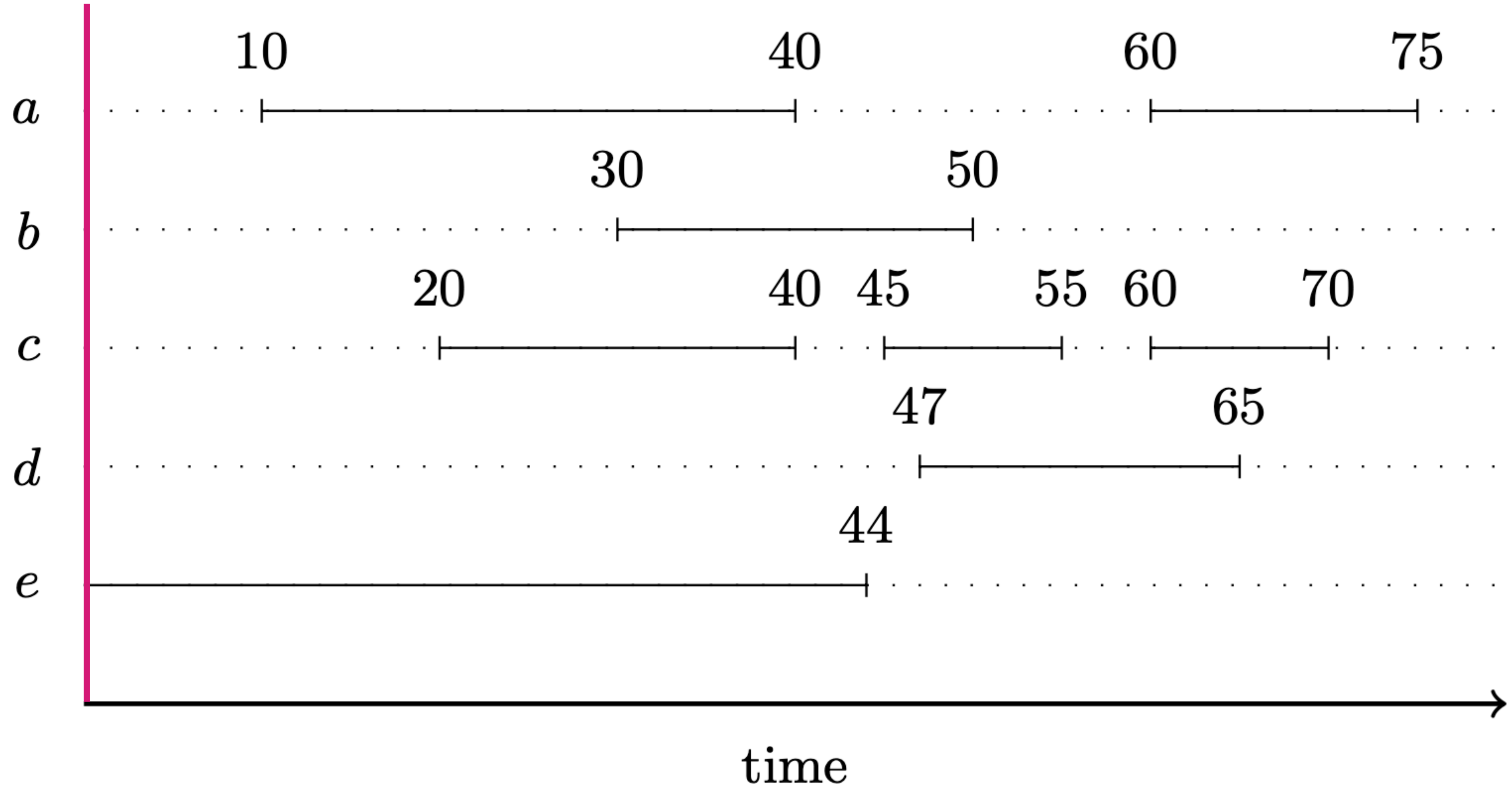




Argumentation Frameworks

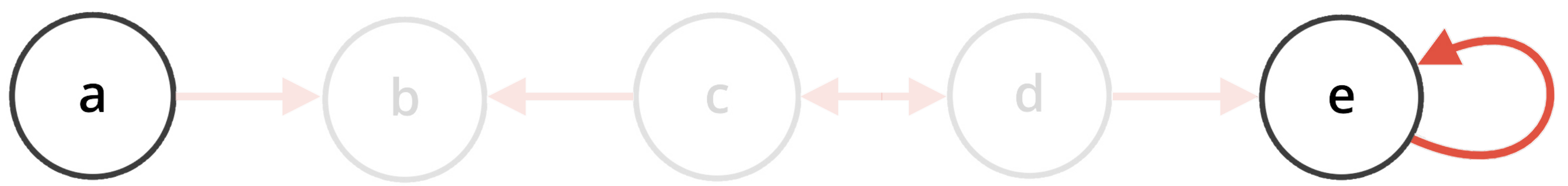
Timed AFs

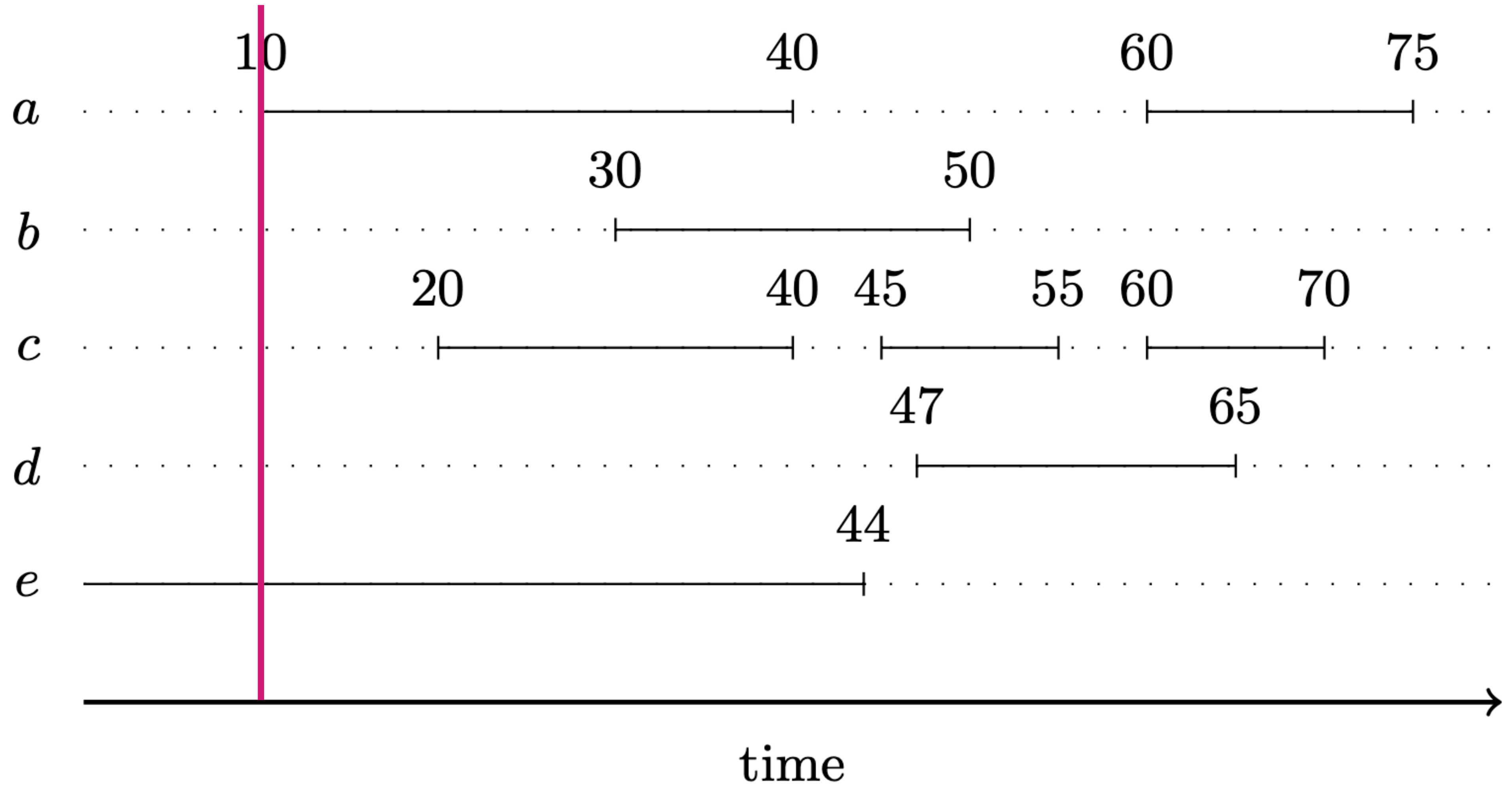




Argumentation Frameworks

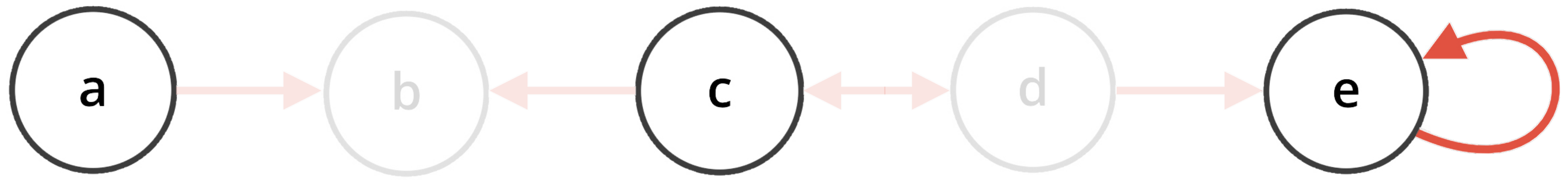
Timed AFs

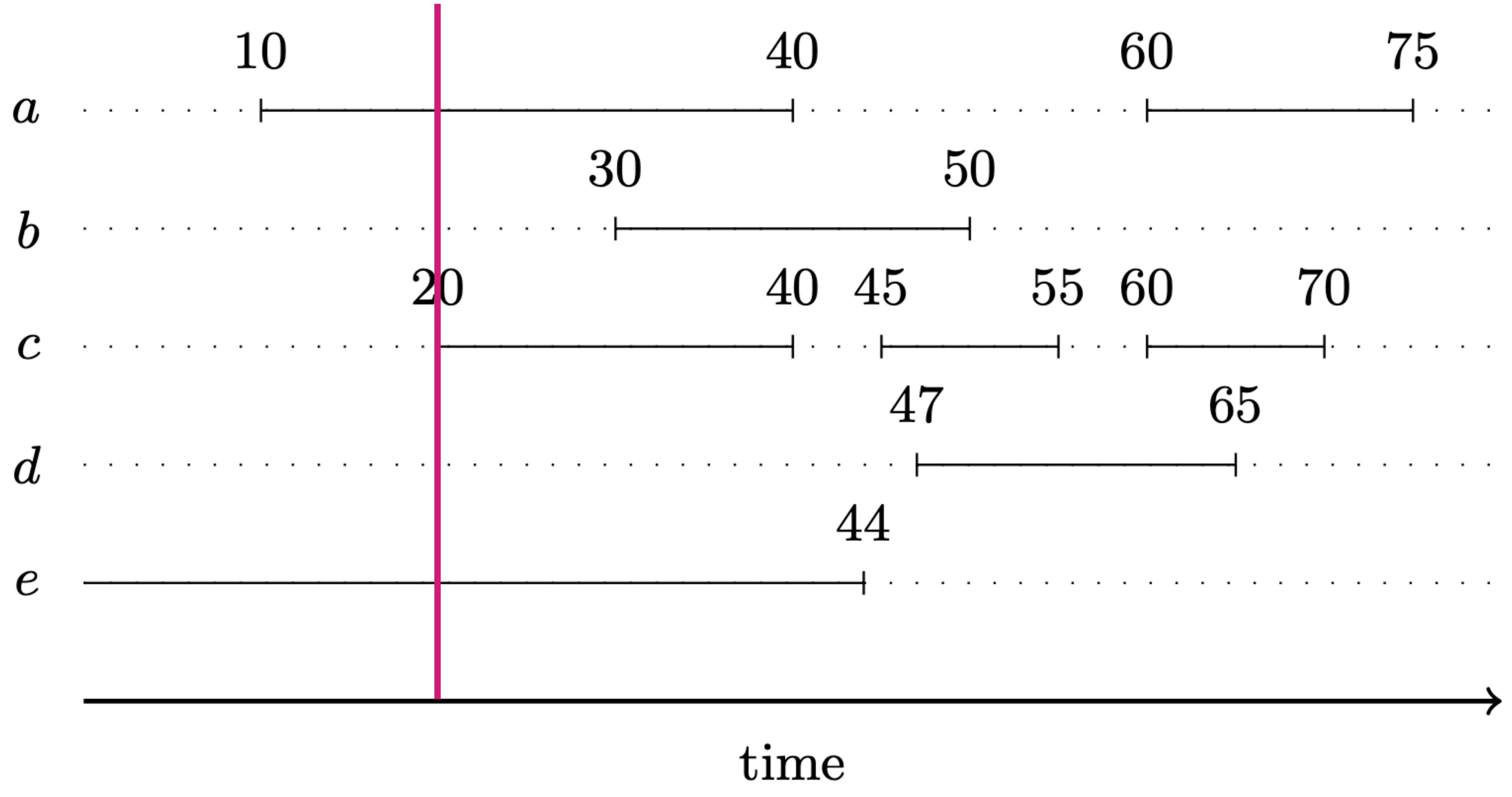




Argumentation Frameworks

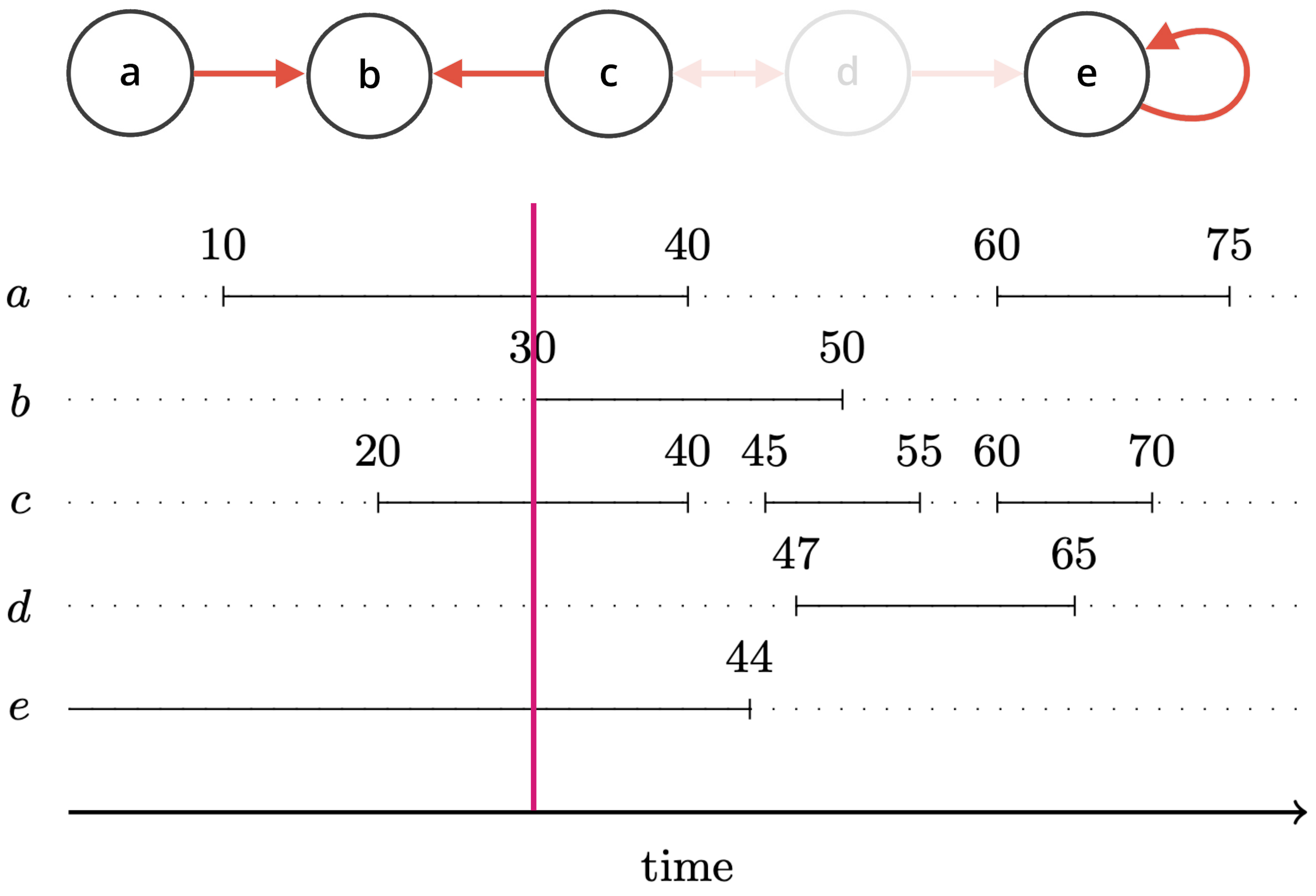
Timed AFs





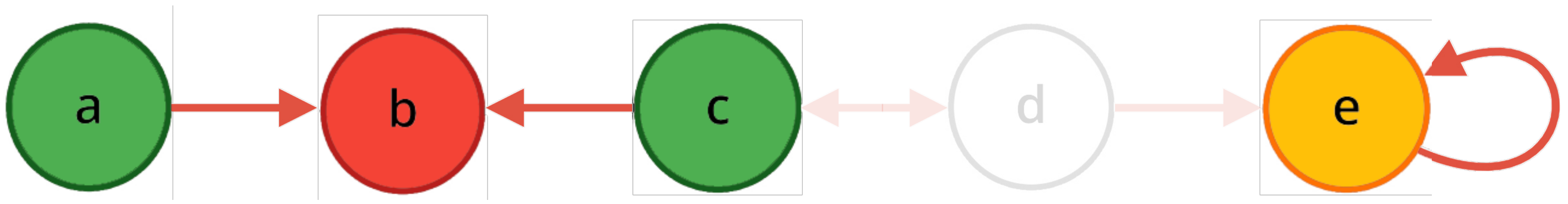
Argumentation Frameworks

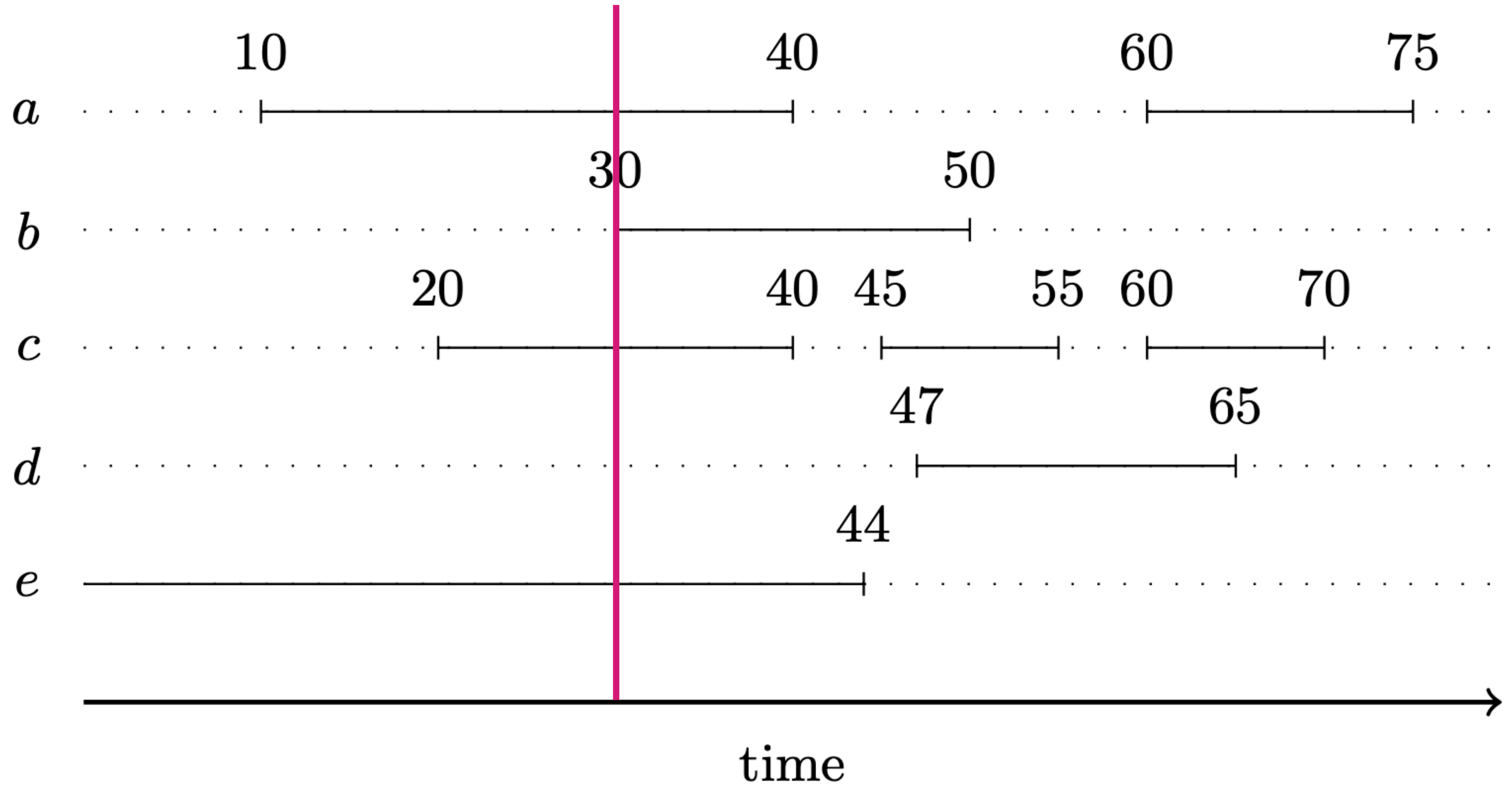
Timed AFs



Argumentation Frameworks

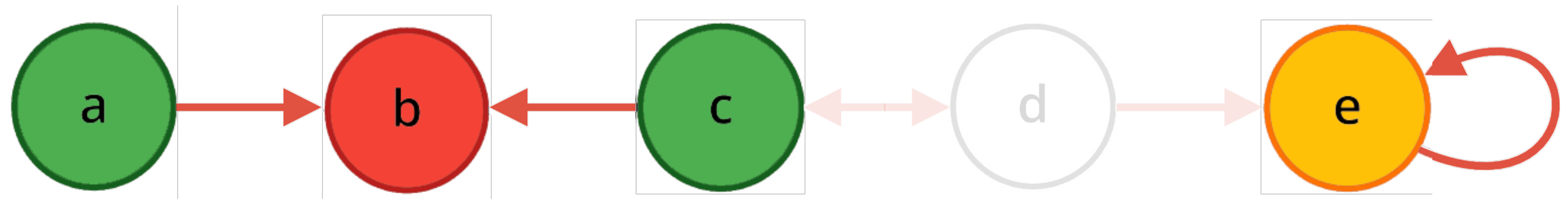
Timed AFs

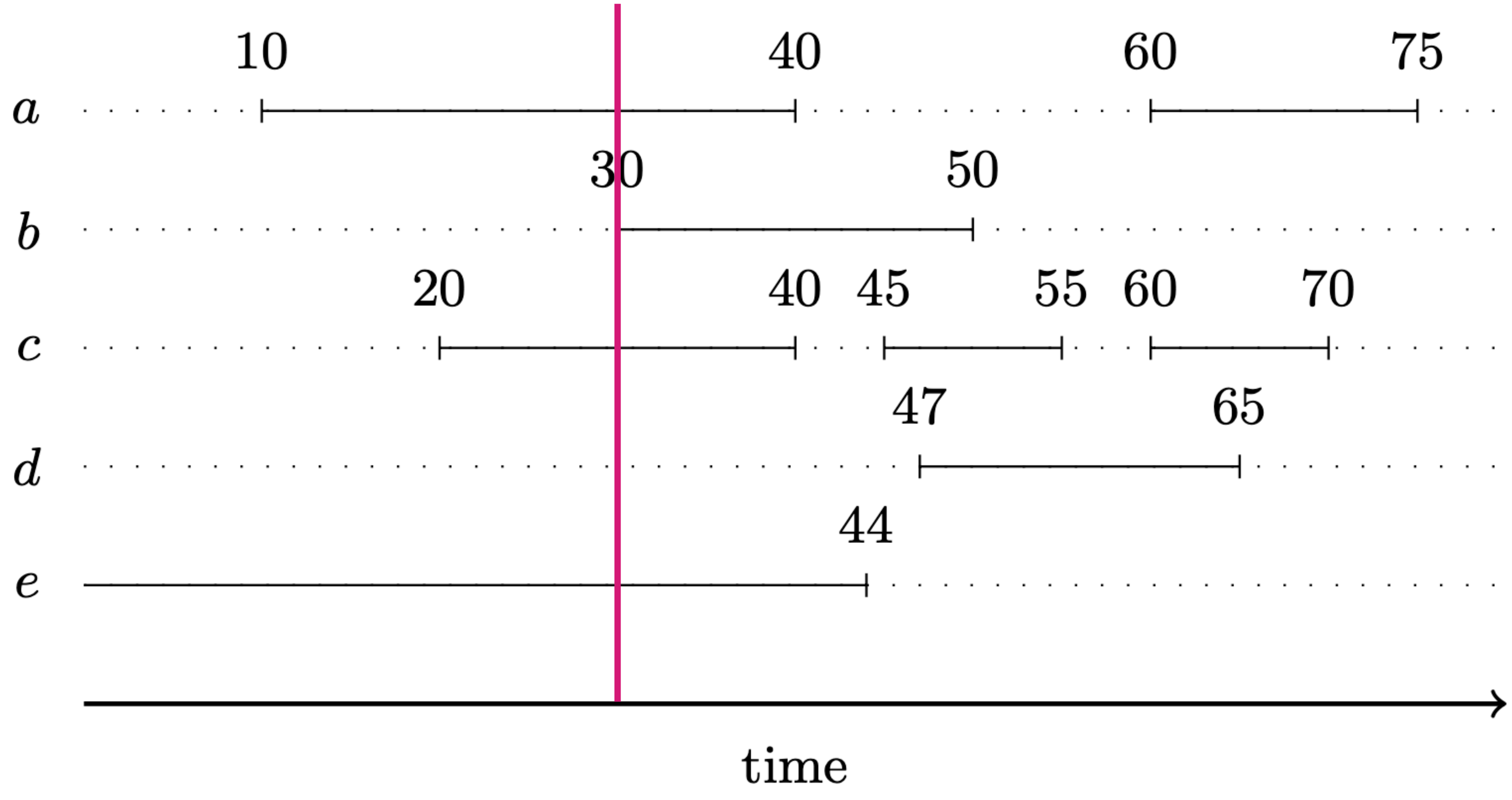




Argumentation Frameworks

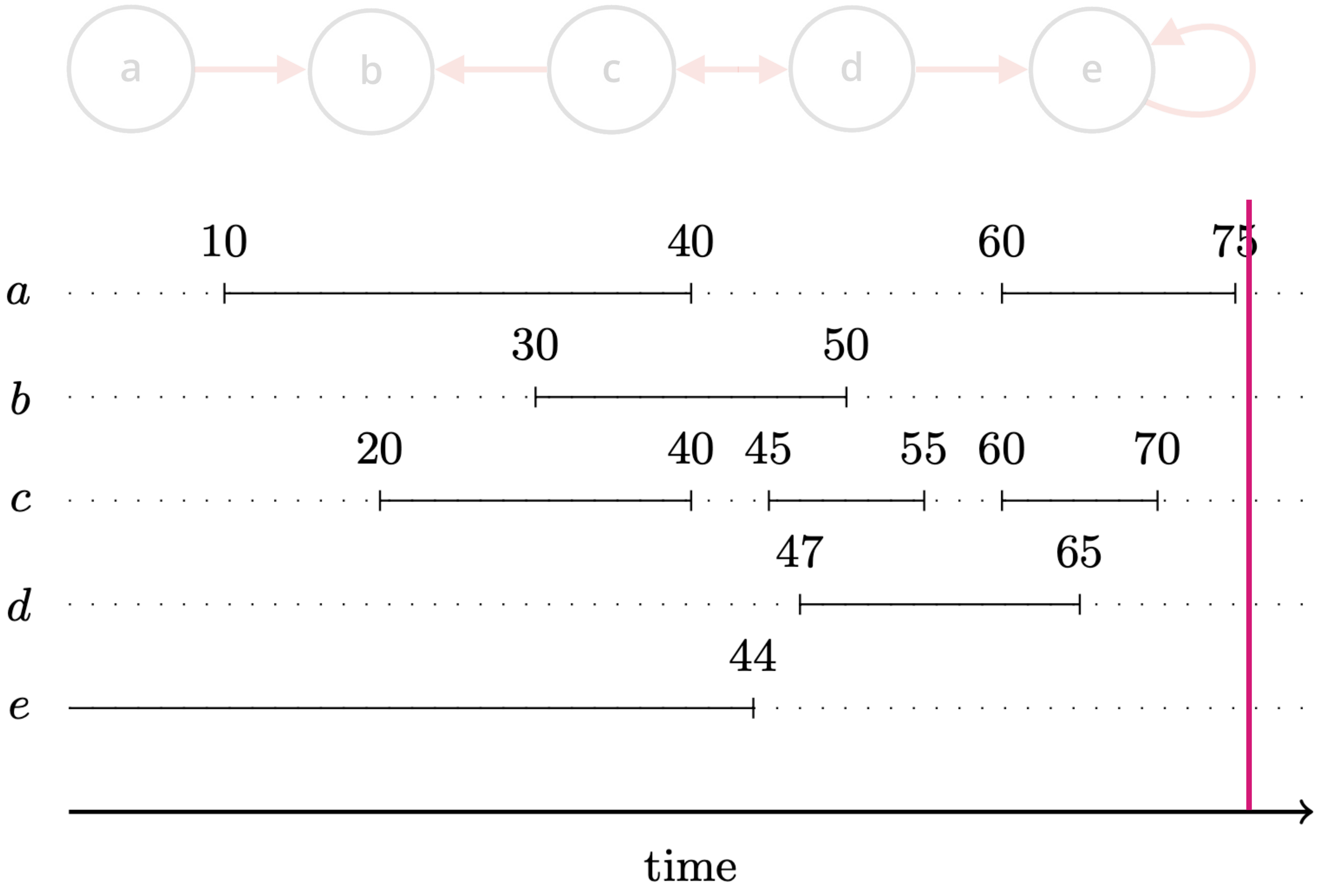
Timed AFs





Argumentation Frameworks

Timed AFs



Timed Concurrent Language for Argumentation

Syntax

- Models concurrent processes taking place over time
- Knowledge base represented through AFs
- Bounded asynchrony and maximal parallelism

- $A ::= success \mid failure \mid add(Arg, R) \rightarrow A \mid rmv(Arg, R) \rightarrow A$
  $\mid A \| A \mid \exists_x A \mid E \mid p(a, l, \sigma, i)$
- $E ::= test_{c,t}(a, l, \sigma) \rightarrow A \mid test_{s,t}(a, l, \sigma) \rightarrow A \mid check_t(Arg, R) \rightarrow A$
  $\mid E + E \mid E +_P E \mid E \|_G E$

Timed Concurrent Language for Argumentation

Operational Semantics (1)

$$\begin{array}{c}\langle add(Arg', R') \rightarrow A, \langle Arg, R\rangle\rangle \longrightarrow \langle A, \langle Arg \cup Arg', R \cup R''\rangle\rangle \\ \text{where } R'' = \{(a,b) \in R' \mid a, b \in Arg \cup Arg'\}\end{array} \qquad \text{Add}$$

$$\begin{array}{c}\langle rmv(Arg', R') \rightarrow A, \langle Arg, R\rangle\rangle \longrightarrow \langle A, \langle Arg \setminus Arg', R \setminus \{R' \cup R''\}\rangle\rangle \\ \text{where } R'' = \{(a,b) \in R \mid a \in Arg' \vee b \in Arg'\}\end{array} \qquad \text{Rmv}$$

$$\frac{Arg' \subseteq Arg \wedge R' \subseteq R \qquad t > 0}{\langle check_t(Arg', R') \rightarrow A, \langle Arg, R\rangle\rangle \longrightarrow \langle A, \langle Arg, R\rangle\rangle} \qquad \text{Ch (1)}$$

$$\frac{Arg' \not\subseteq Arg \vee R' \not\subseteq R \qquad t > 0}{\langle check_t(Arg', R') \rightarrow A, \langle Arg, R\rangle\rangle \longrightarrow \langle check_{t-1}(Arg', R') \rightarrow A, \langle Arg, R\rangle\rangle} \qquad \text{Ch (2)}$$

$$\langle check_0(Arg', R') \rightarrow A, F\rangle \longrightarrow \langle failure, F\rangle \qquad \text{Ch (3)}$$

Timed Concurrent Language for Argumentation

Operational Semantics (2)

$$\frac{\exists L \in S_\sigma(F) \mid l \in L(a) \quad t > 0}{\langle test_{c,t}(a,l,\sigma) \rightarrow A, F\rangle \longrightarrow \langle A, F\rangle} \qquad \text{CT (1)}$$

$$\frac{\forall L \in S_\sigma(F).l \notin L(a) \quad t > 0}{\langle test_{c,t}(a,l,\sigma) \rightarrow A, F\rangle \longrightarrow \langle test_{c,t-1}(a,l,\sigma) \rightarrow A, F\rangle} \qquad \text{CT (2)}$$

$$\langle test_{c,0}(a,l,\sigma) \rightarrow A, F\rangle \longrightarrow \langle failure, F\rangle \qquad \text{CT (3)}$$

$$\frac{\forall L \in S_\sigma(F).l \in L(a) \quad t > 0}{\langle test_{s,t}(a,l,\sigma) \rightarrow A, F\rangle \longrightarrow \langle A, F\rangle} \qquad \text{ST (1)}$$

$$\frac{\exists L \in S_\sigma(F) \mid l \notin L(a) \quad t > 0}{\langle test_{s,t}(a,l,\sigma) \rightarrow A, F\rangle \longrightarrow \langle test_{s,t-1}(a,l,\sigma) \rightarrow A, F\rangle} \qquad \text{ST (2)}$$

$$\langle test_{s,0}(a,l,\sigma) \longrightarrow A, F\rangle \longrightarrow \langle failure, F\rangle \qquad \text{ST (3)}$$

Timed Concurrent Language for Argumentation

Operational Semantics (3)

$$\frac{\langle A_1, F\rangle \longrightarrow \langle A_1', F'\rangle, \quad \langle A_2, F\rangle \longrightarrow \langle A_2', F''\rangle}{\langle A_1 \| A_2, F\rangle \longrightarrow \langle A_1' \| A_2', *(F, F', F'')\rangle} \qquad \text{Pa}$$

$$\frac{\langle E_1, F\rangle \longrightarrow \langle A_1, F\rangle, \ \langle E_2, F\rangle \longrightarrow \langle A_2, F\rangle, \quad E_1, E_2 \notin \mathcal{E}_0, \quad A_1 \notin \mathcal{E}}{\langle E_1 \|_G E_2, F\rangle \longrightarrow \langle A_1 \| A_2, F\rangle} \qquad \text{GP (1)}$$

$$\frac{\langle E_1, F\rangle \longrightarrow \langle E_1', F\rangle, \ \langle E_2, F\rangle \longrightarrow \langle E_2', F\rangle, \quad E_1, E_2 \notin \mathcal{E}_0, \quad E_1', E_2' \in \mathcal{E}}{\langle E_1 \|_G E_2, F\rangle \longrightarrow \langle E_1' \|_G E_2', F\rangle} \qquad \text{GP (2)}$$

$$\frac{E_1 \in \mathcal{E}_0, \langle E_2, F\rangle \longrightarrow \langle A_2, F\rangle}{\langle E_1 \|_G E_2, F\rangle \longrightarrow \langle A_2, F\rangle} \qquad \text{GP (3)}$$

Timed Concurrent Language for Argumentation

Operational Semantics (4)

$$\frac{\langle E_1, F\rangle \longrightarrow \langle A_1, F\rangle, \quad E_1 \notin \mathcal{E}_0, \quad A_1 \notin \mathcal{E}}{\langle E_1 + E_2, F\rangle \longrightarrow \langle A_1, F\rangle} \qquad \frac{E_1 \in \mathcal{E}_0, \langle E_2, F\rangle \longrightarrow \langle A_2, F\rangle}{\langle E_1 + E_2, F\rangle \longrightarrow \langle A_2, F\rangle} \text{ ND (1)}$$

$$\frac{\langle E_1, F\rangle \longrightarrow \langle E_1', F\rangle, \; \langle E_2, F\rangle \longrightarrow \langle E_2', F\rangle, \quad E_1, E_2 \notin \mathcal{E}_0, \quad E_1', E_2' \in \mathcal{E}}{\langle E_1 + E_2, F\rangle \longrightarrow \langle E_1' + E_2', F\rangle} \qquad \text{ND (2)}$$

$$\frac{\langle E_1, F\rangle \longrightarrow \langle A_1, F\rangle, \quad E_1 \notin \mathcal{E}_0, \quad A_1 \notin \mathcal{E}}{\langle E_1 +_P E_2, F\rangle \longrightarrow \langle A_1, F\rangle} \qquad \text{If (1)}$$

$$\frac{\langle E_1, F\rangle \longrightarrow \langle E_1', F\rangle, \quad E_1 \notin \mathcal{E}_0, \quad E_1' \in \mathcal{E}}{\langle E_1 +_P E_2, F\rangle \longrightarrow \langle E_1' +_P E_2, F\rangle} \qquad \frac{E_1 \in \mathcal{E}_0, \langle E_2, F\rangle \longrightarrow \langle A_2, F\rangle}{\langle E_1 +_P E_2, F\rangle \longrightarrow \langle A_2, F\rangle} \qquad \text{If (2)}$$

Modelling Timed AFs

Example

- Each agent handles one argument
- Agent_a: $sleep(9) \rightarrow (add(\{a\},\{(b,a),(d,a)\}) \rightarrow (sleep(30) \rightarrow (rmv(\{a\},\{\}) \rightarrow (sleep(18) \rightarrow (add(\{a\},\{(b,a),(d,a)\}) \rightarrow (sleep(15) \rightarrow (rmv(\{a\},\{\}) \rightarrow success))))))$
- Agent_b: $sleep(29) \rightarrow (add(\{b\},\{(b,a),(c,b)\}) \rightarrow (sleep(20) \rightarrow (rmv(\{b\},\{\}) \rightarrow success)))$
- Agent_a || Agent_b

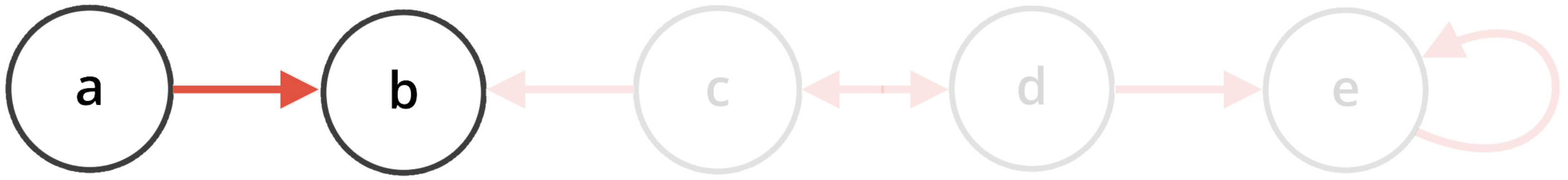
$sleep(t) \rightarrow A$ is a shortcut for

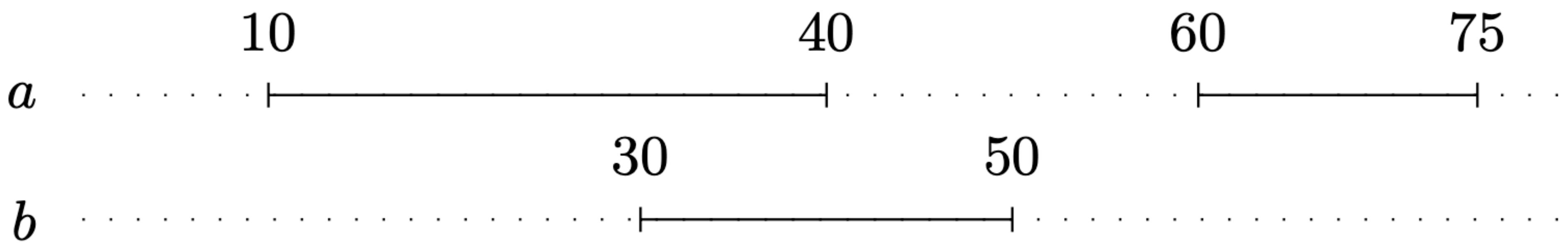
$$\begin{cases} A & \text{if } t \leq 0 \\ check_1(\{\},\{\}) \rightarrow (sleep(t-1) \rightarrow A) & \text{otherwise} \end{cases}$$

Modelling Timed AFs

Example

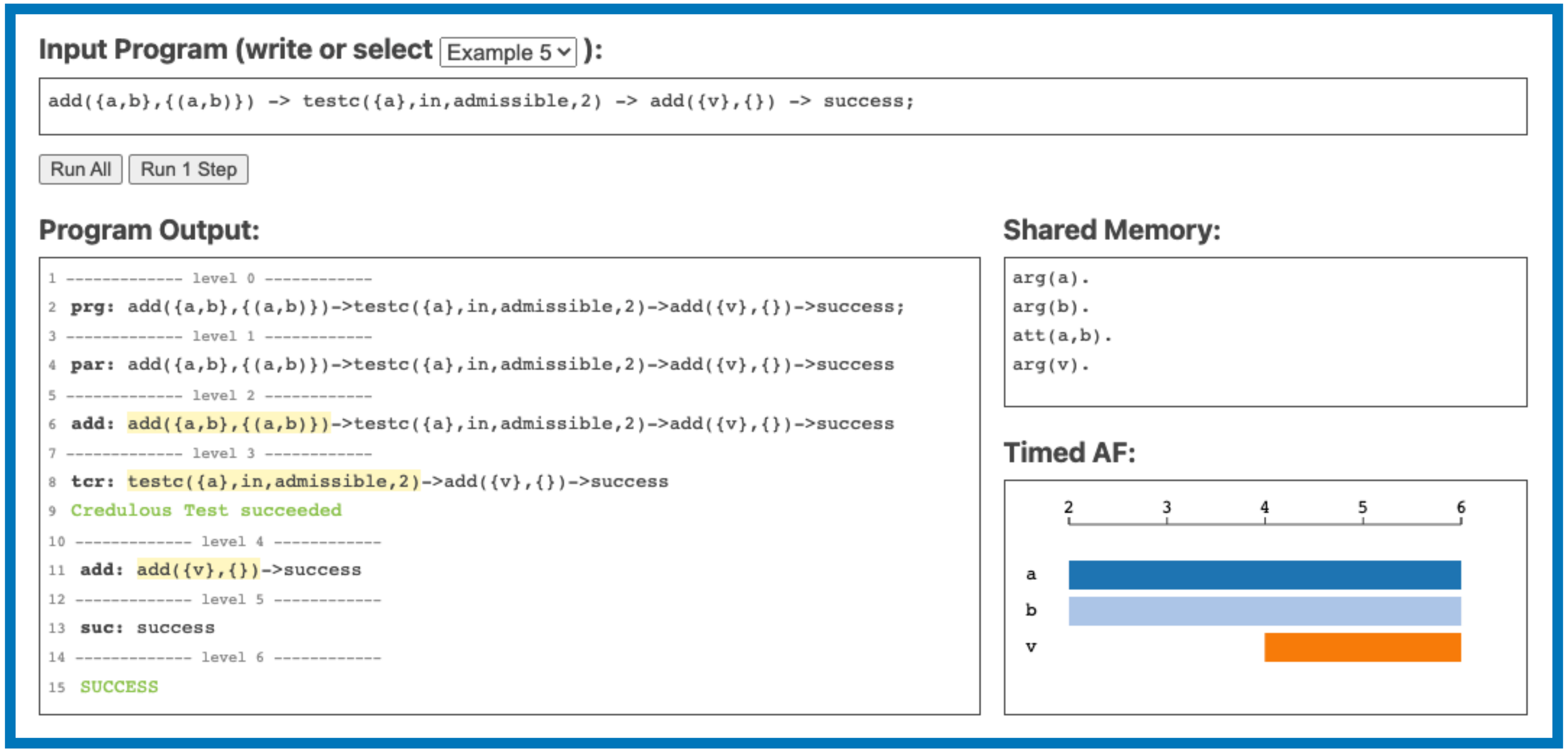
- Each agent handles one argument
- Agent_a: $sleep(9) \rightarrow (add(\{a\}, \{(b,a),(d,a)\}) \rightarrow (sleep(30) \rightarrow (rmv(\{a\},\{\}) \rightarrow (sleep(18) \rightarrow (add(\{a\}, \{(b,a),(d,a)\}) \rightarrow (sleep(15) \rightarrow (rmv(\{a\},\{\}) \rightarrow success)))))))$
- Agent_b: $sleep(29) \rightarrow (add(\{b\}, \{(b,a),(c,b)\}) \rightarrow (sleep(20) \rightarrow (rmv(\{b\},\{\}) \rightarrow success)))$
- Agent_a || Agent_b





tcla Web Interface

- http://conarg.dmi.unipg.it/tcla
- Run all / step-by-step execution
- Shows program output + shared memory status + Timed AF



```
add({a,b},{(a,b)}) -> testc({a},in,admissible,2) -> add({v},{}) -> success;
```

```
------------- level 0 ------------
prg: add({a,b},{(a,b)})->testc({a},in,admissible,2)->add({v},{})->success;
------------- level 1 ------------
par: add({a,b},{(a,b)})->testc({a},in,admissible,2)->add({v},{})->success
------------- level 2 ------------
add: add({a,b},{(a,b)})->testc({a},in,admissible,2)->add({v},{})->success
------------- level 3 ------------
tcr: testc({a},in,admissible,2)->add({v},{})->success
Credulous Test succeeded
------------- level 4 ------------
add: add({v},{})->success
------------- level 5 ------------
suc: success
------------- level 6 ------------
SUCCESS
```

```
arg(a).
arg(b).
att(a,b).
arg(v).
```

Conclusion

- `tcla` models concurrent argumentation processes over time

Conclusion

- `tcla` models concurrent argumentation processes over time
- We showed how to translate Timed AFs into `tcla` processes

Conclusion

- `tcla` models concurrent argumentation processes over time
- We showed how to translate Timed AFs into `tcla` processes
    - the arguments in the store at time t are all and only the arguments available at t in the original Timed AF

Conclusion

- `tcla` models concurrent argumentation processes over time
- We showed how to translate Timed AFs into `tcla` processes
  - the arguments in the store at time t are all and only the arguments available at t in the original Timed AF
  - the set of accepted arguments in the store at t coincides with the set of extensions at t in the original Timed AF

Conclusion

- `tcla` models concurrent argumentation processes over time
- We showed how to translate Timed AFs into `tcla` processes
  - the arguments in the store at time t are all and only the arguments available at t in the original Timed AF
  - the set of accepted arguments in the store at t coincides with the set of extensions at t in the original Timed AF
- Working implementation available online

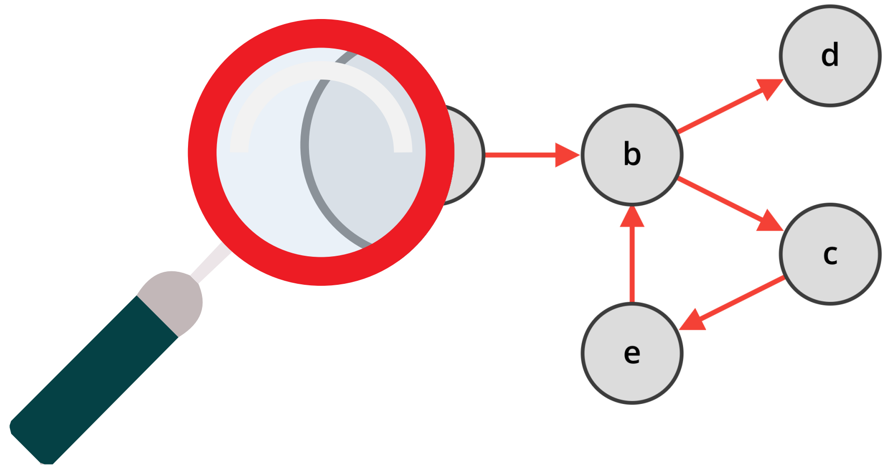
Conclusion

- `tcla` models concurrent argumentation processes over time
- We showed how to translate Timed AFs into `tcla` processes
  - the arguments in the store at time t are all and only the arguments available at t in the original Timed AF
  - the set of accepted arguments in the store at t coincides with the set of extensions at t in the original Timed AF
- Working implementation available online

… `tcla` can do more than just modelling Timed AFs!

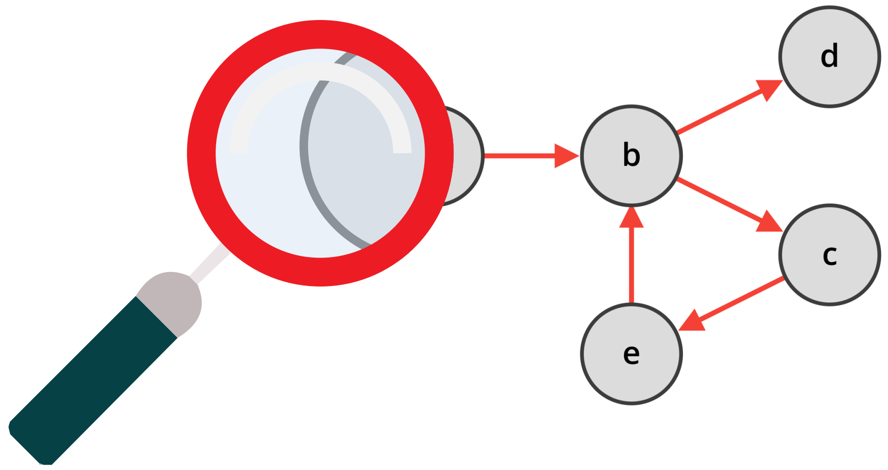
Future Work

- Negotiation/debating/persuasion between agents



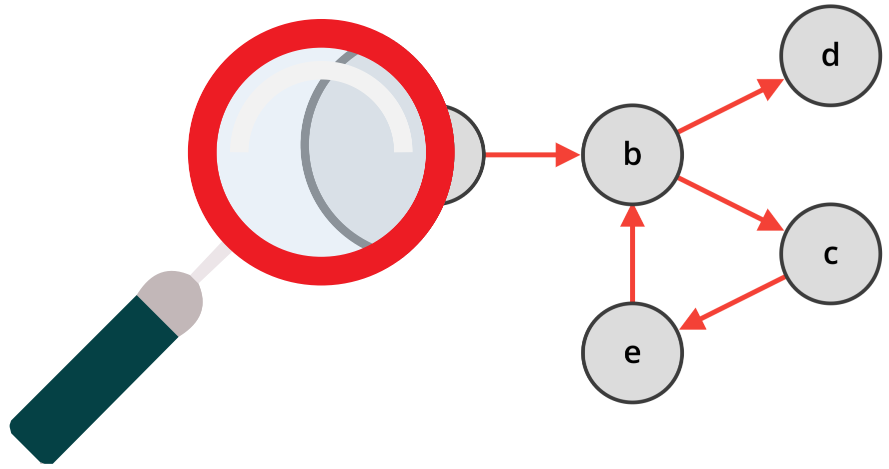
Future Work

- Negotiation/debating/persuasion between agents
- Use interleaving instead of maximal parallelism for time passing



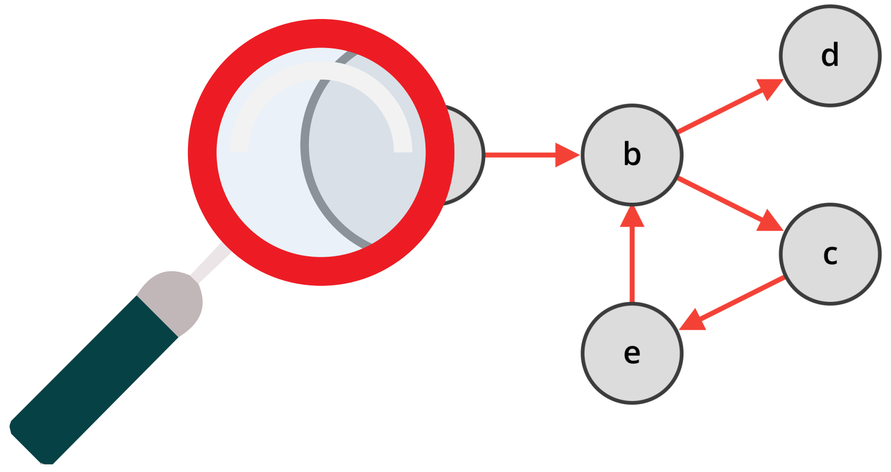
Future Work

- Negotiation/debating/persuasion between agents
- Use interleaving instead of maximal parallelism for time passing
- Study time-dependent notions of acceptability



Future Work

- Negotiation/debating/persuasion between agents
- Use interleaving instead of maximal parallelism for time passing
- Study time-dependent notions of acceptability
- Connections with the AGM Framework



Thank you for your attention!

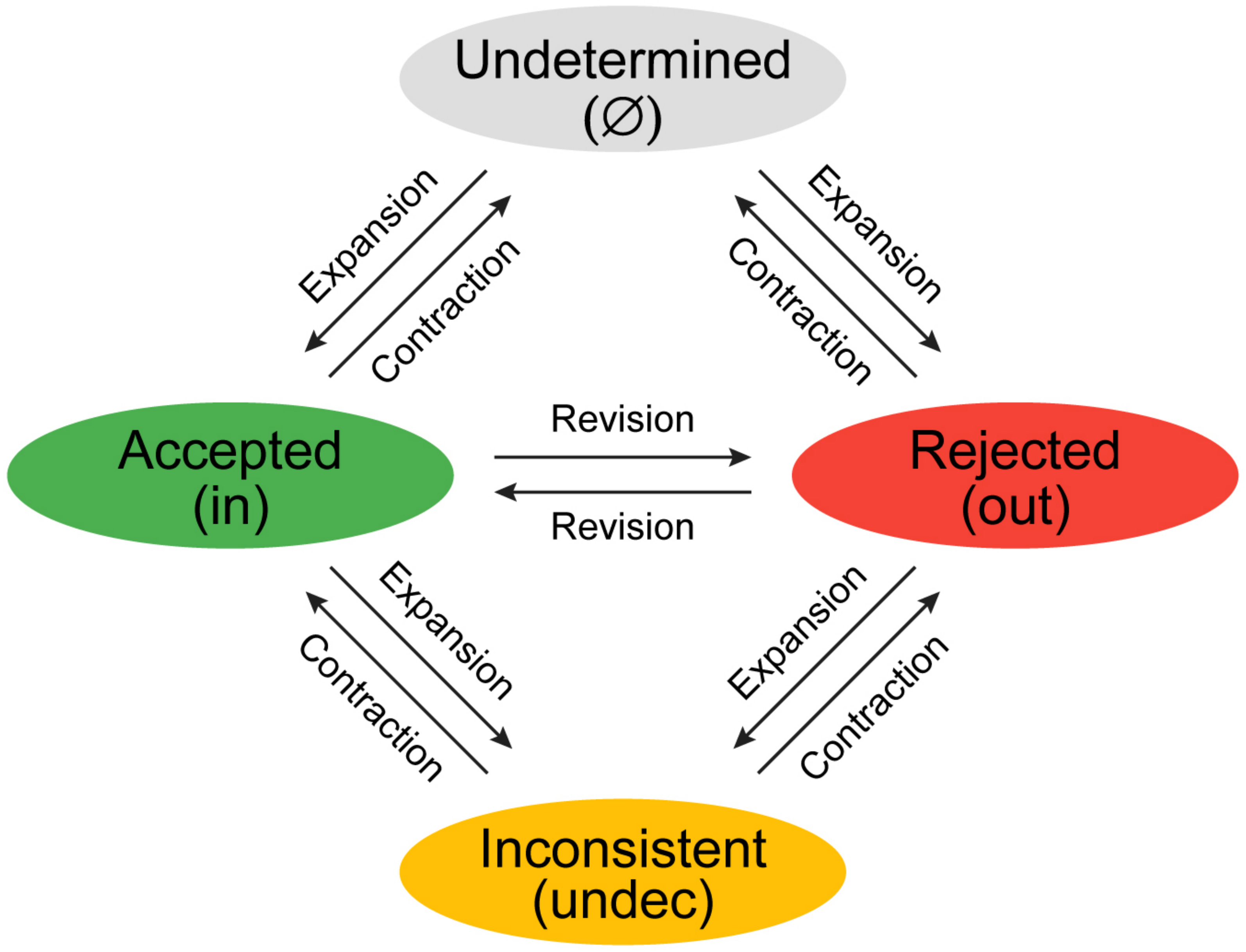
UNIVERSITÀ DEGLI STUDI
DI PERUGIA

Università degli Studi "G. d'Annunzio"

Timed Concurrent Language for Argumentation

Stefano Bistarelli, Maria Chiara Meo, <u>Carlo Taticchi</u>

CILC 2021


Undetermined (∅)
Expansion
Contraction
Expansion
Contraction
Accepted (in)
Revision
Revision
Rejected (out)
Expansion
Contraction
Expansion
Contraction
Inconsistent (undec)