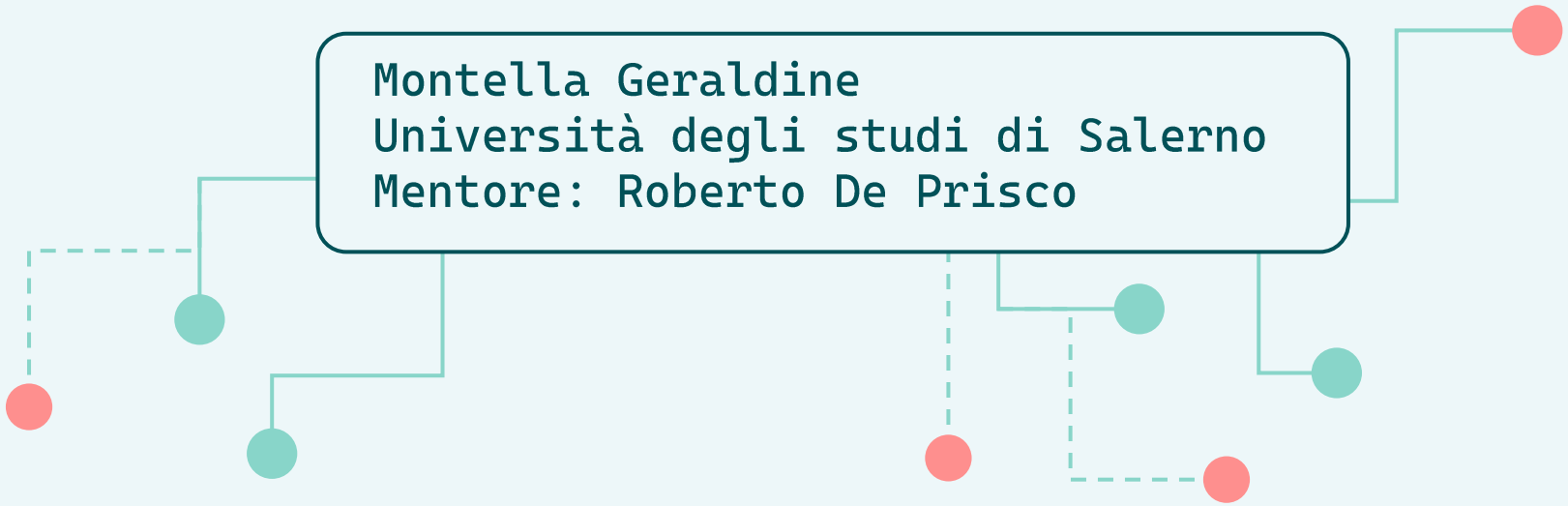
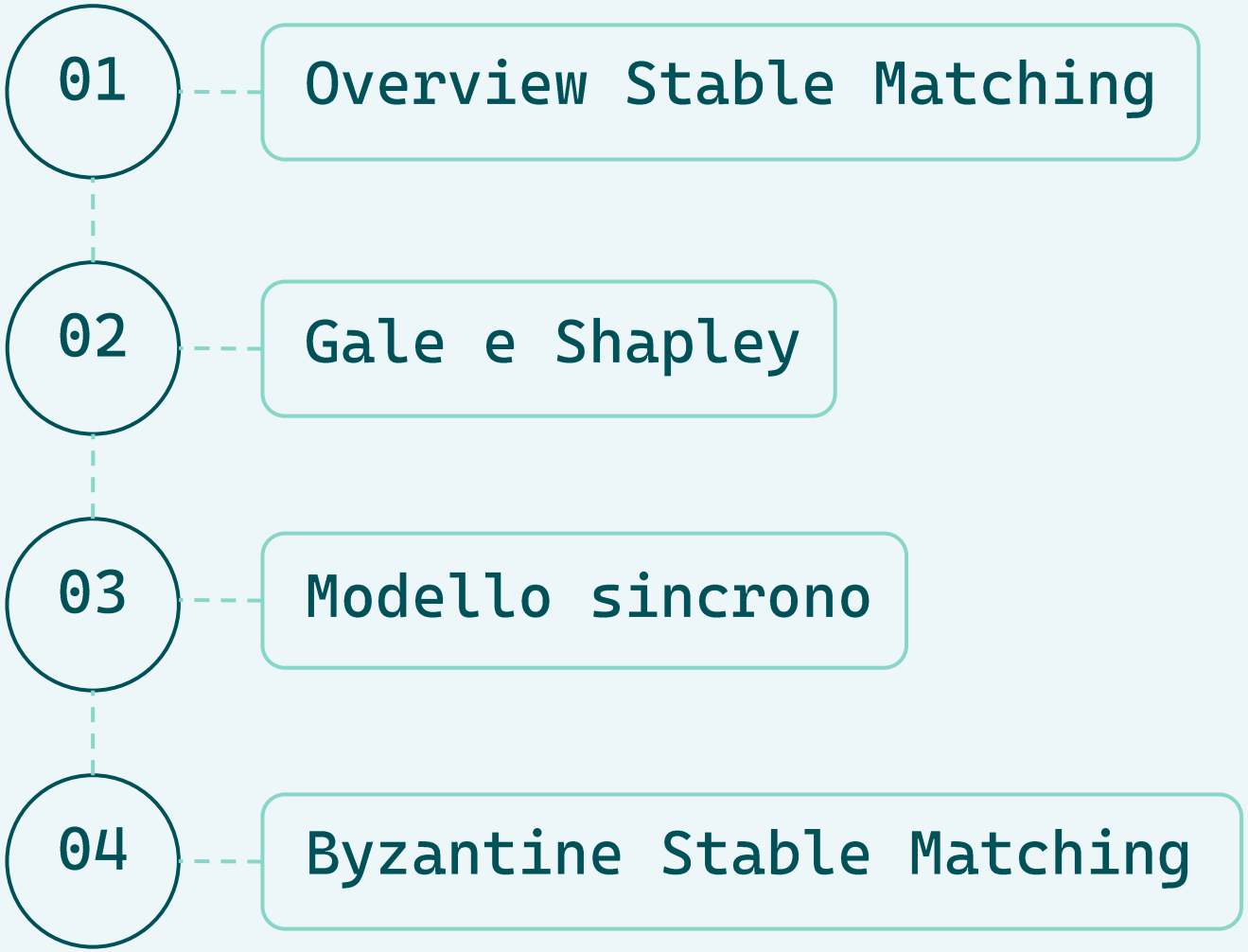
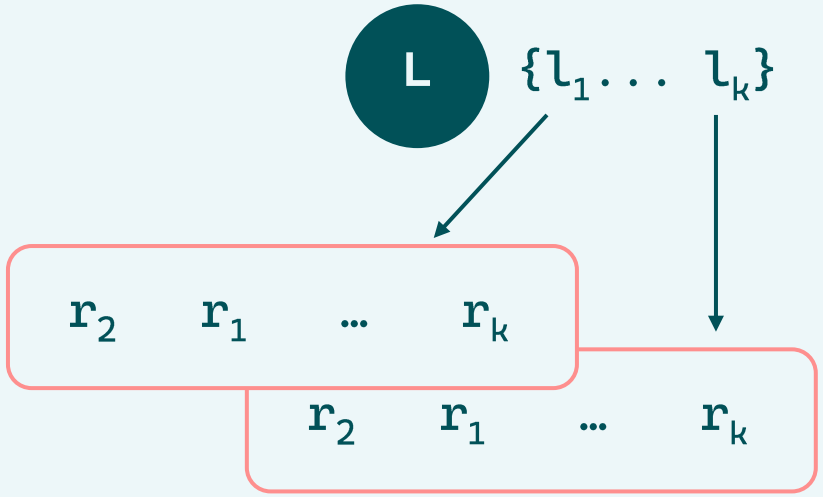
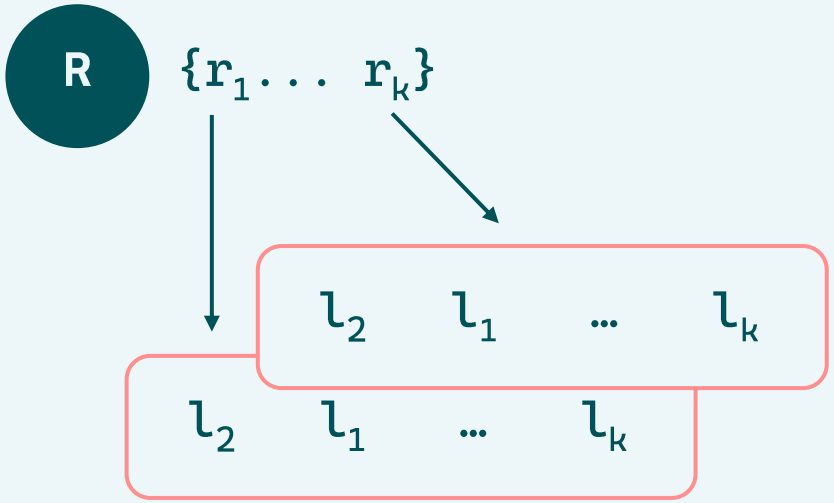
Stable Matching: dal modello classico al Byzantine Stable Matching

Montella Geraldine
Università degli studi di Salerno
Mentore: Roberto De Prisco




01
Overview Stable Matching
02
Gale e Shapley
03
Modello sincrono
04
Byzantine Stable Matching


L
{l1... lk}
r2 r1 … rk
r2 r1 … rk

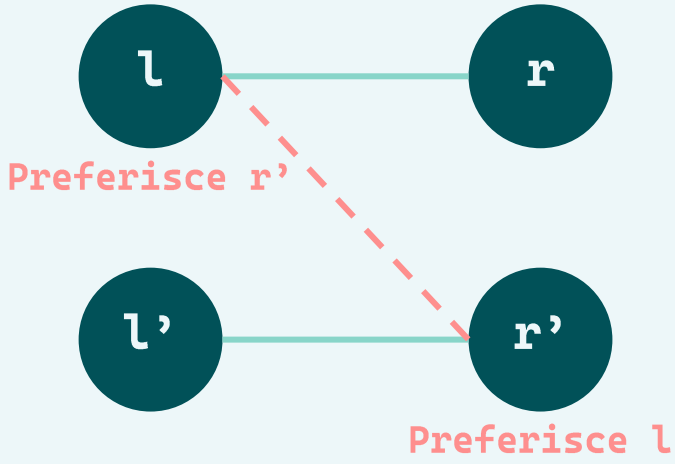

R
{r1... rk}
l2 l1 … lk
l2 l1 … lk

Un **matching M** è $M \subseteq L \times R$ tale che:

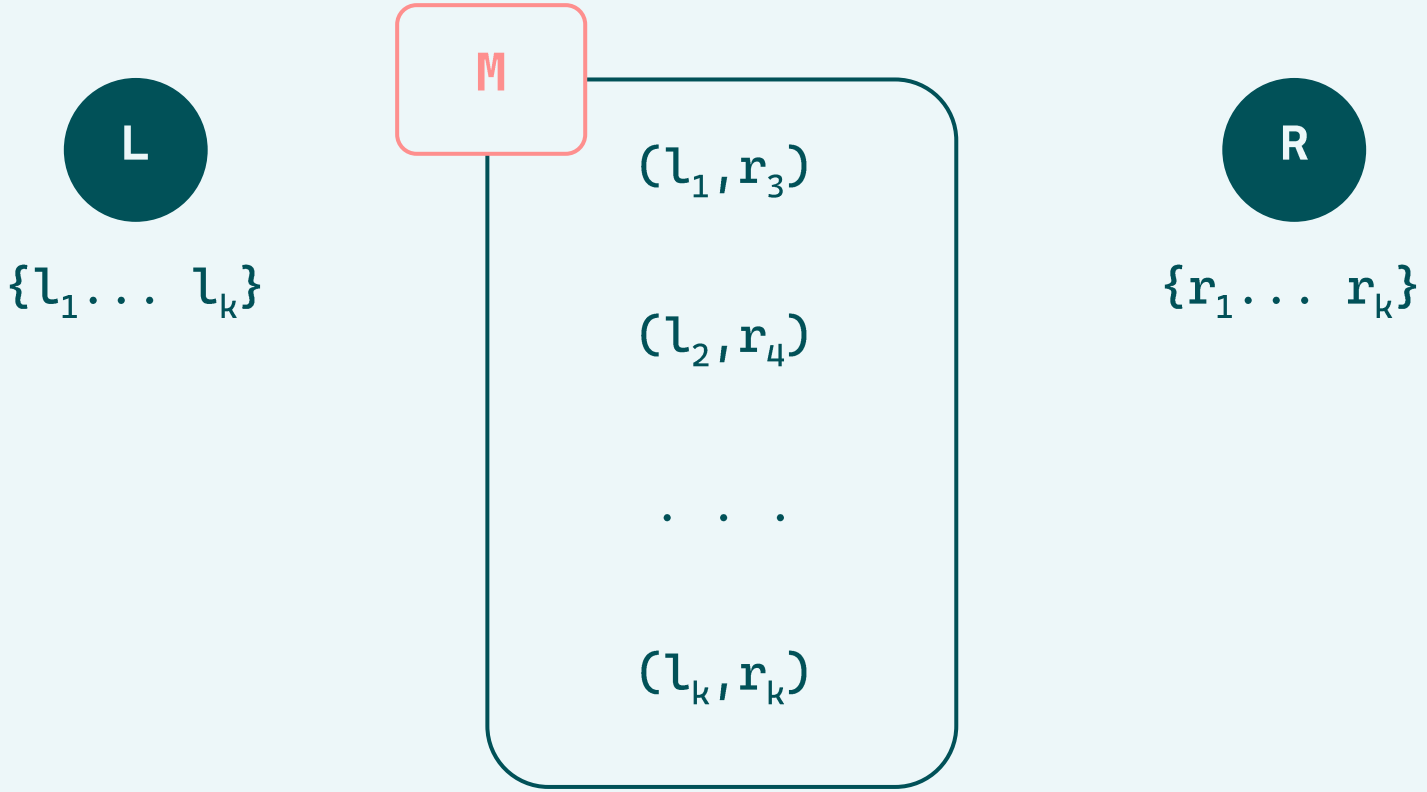
Condizione di univocità
Per ogni coppia (x_h, y_k) $(x_i, y_j) \in M$
e risulta $x_h \neq x_i$ e $y_k \neq y_j$

Matching M si dice **completo** quando
tutti gli elementi sono accoppiati

Un **matching M** è detto stabile se privo di coppie se privo di **coppie bloccanti**, se considero due coppie $(l,r),(l',r') \in M$



Obiettivo:
Trovare un matching stabile



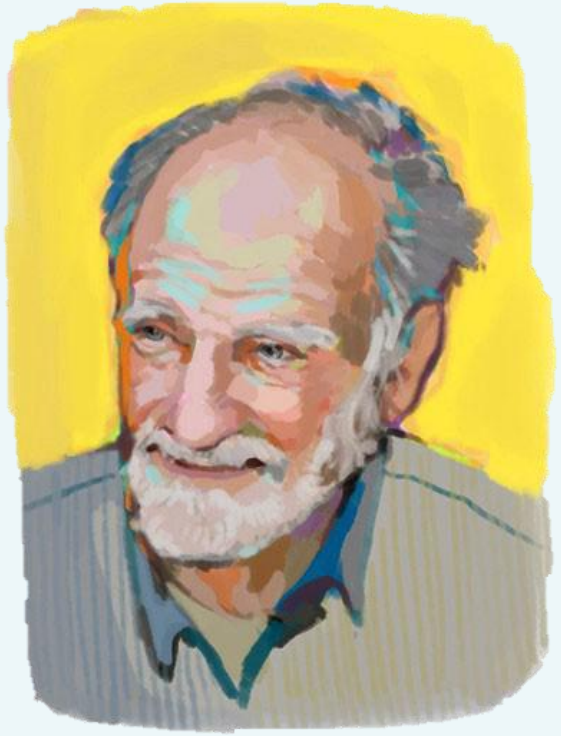
Content Delivery Network (CDN):

Abbinare gruppi di client a cluster di server per un bilanciamento del carico efficiente e robusto .

Rete Wireless:

Assegnare canali o risorse radio agli utenti. La stabilità garantisce un uso efficiente dello spettro e riduce le interferenze.

David Gale
Professor, UC Berkeley



Llyod Shapley
Professor Emeritus, UCLA

Sito refernce:
https://www.universityofcalifornia.edu/news/how-two-matchmakers-won-nobel-prize

Esiste sempre un matching stabile?

Teorema dell'Esistenza

$L \longrightarrow \{l_1 \dots l_k\}$ $\quad$ $R \longrightarrow \{r_1 \dots r_k\}$

Abbiamo che $|L| = |R| = k$, e date le liste di preferenza complete e strettamente ordinate, esiste **sempre almeno uno stable matching.**

Deffered Acceptance

let all l e r be unmached

Repeat

unmatched l proposes to the most preferred r in his list

if r is unmacheted **then**

match r with l

else if r is matched to l', but she prefers l to l' **then**

match r to l and leave l' unmatched

else

r removes l form his preference list

end if

Until ogni l has a match or has reached the end of his list

Return matching M

Deffered Acceptance

```
let all l e r be unmached
Repeat
    unmatched l proposes to the most preferred r in his list
    if r is unmacheted then
          match r with l
    else if r is matched to r', but she prefers l to l' then
          match r to l and leave l' unmatched
    else
          r removes l form his preference list
    end if
Until ogni l has a match or has reached the end of his list
Return matching M
```

Accettazione differita
– Si assicura la possibilità di trovare un match migliore

Deffered Acceptance

```
let all l e r be unmached
Repeat
    unmatched l proposes to the most preferred r in his list
    if r is unmacheted then
          match r with l
    else if r is matched to r', but she prefers l to l' then
          match r to l and leave l' unmatched
    else
          r removes l form his preference list
    end if
Until ogni l has a match or has reached the end of his list
Return matching M
```

Possiamo affermare che:

- **Termina**

Abbiamo che $|L| = |R| = k$ nel caso pessimo si richiederanno k^2 passi

Deffered Acceptance

let all l e r be unmached

Repeat

unmatched l proposes to the most preferred r in his list

if r is unmacheted **then**

match r with l

else if r is matched to r', but she prefers l to l' **then**

match r to l and leave l' unmatched

else

r removes l form his preference list

end if

Until ogni l has a match or has reached the end of his list

Return matching M

Possiamo affermare che:

- **Termina**
- **Il matching ottenuto è di sicuro è completo**

Deffered Acceptance

```
let all l e r be unmached
Repeat
    unmatched l proposes to the most preferred r in his list
    if r is unmacheted then
          match r with l
    else if r is matched to r', but she prefers l to l' then
          match r to l and leave l' unmatched
    else
          r removes l form his preference list
    end if
Until ogni l has a match or has reached the end of his list
Return matching M
```

Possiamo affermare che:

- **Termina**
- **Il matching ottenuto è di sicuro è completo**
- **Non ci sono coppie bloccanti perciò il matching è stabile**

Non ci sono coppie bloccanti perciò matching è stabile

Dimostrazione per assurdo

Supponiamo l'esistenza di una coppia bloccante
(l,r)
In cui l preferisce r al suo attuale match
- Se r preferisse l si sarebbe proposto prima

- Se r e l non sono in match, vuol dire che r lo ha rifiutato per un match migliore (r può solo migliorare)

Ma dato che r può solo migliorare il match finale deve essere equivalente o migliore. Si contraddice l'ipotesi iniziale

Non ci sono coppie bloccanti perciò matching è stabile

Dimostrazione per assurdo

Proposer-Optimal

Supponiamo l'esistenza di una coppia bloccante
(l,r)
In cui l preferisce r al suo attuale match
- Se r preferisse **l si sarebbe proposto prima**

- Se r e l non sono in match, vuol dire che r lo ha rifiutato per un match migliore (r può solo migliorare)

Ma dato che r può solo migliorare il match finale deve essere equivalente o migliore. Si contraddice l'ipotesi iniziale

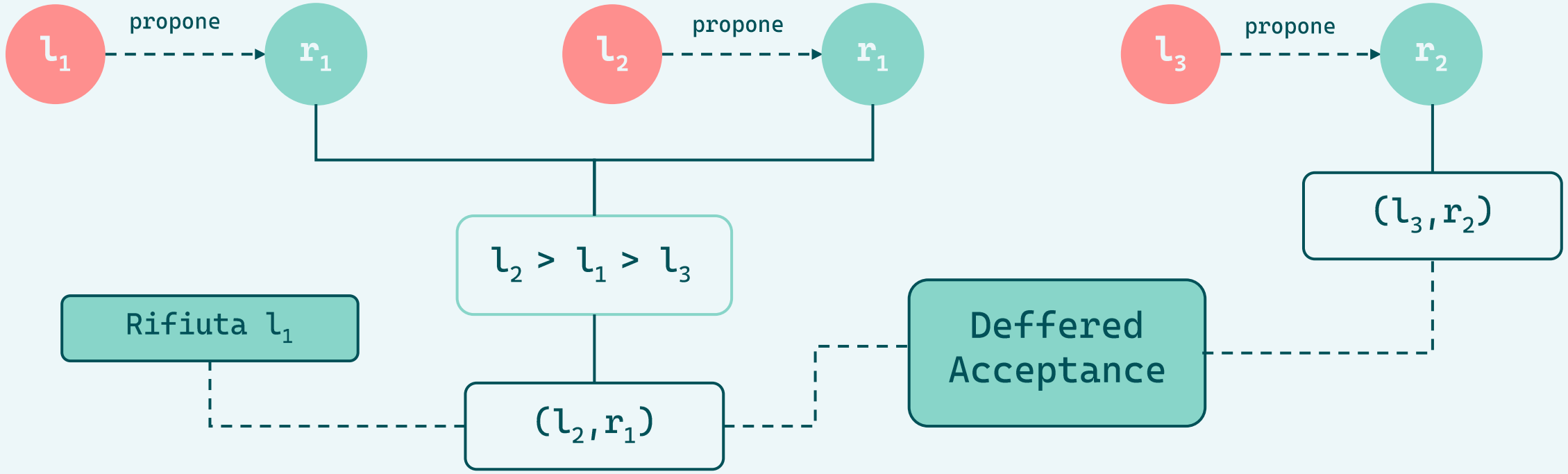
Esempio pratico per l'esecuzione dello pseudocodice di **Gale-Shapley**

L $\{l_1, l_2, l_3\}$

R $\{r_1, r_2, r_3\}$

l_1	$r_1 > r_2 > r_3$	$l_2 > l_1 > l_3$	r_1
l_2	$r_1 > r_2 > r_3$	$l_1 > l_3 > l_2$	r_2
l_3	$r_2 > r_1 > r_3$	$l_1 > l_2 > l_3$	r_3

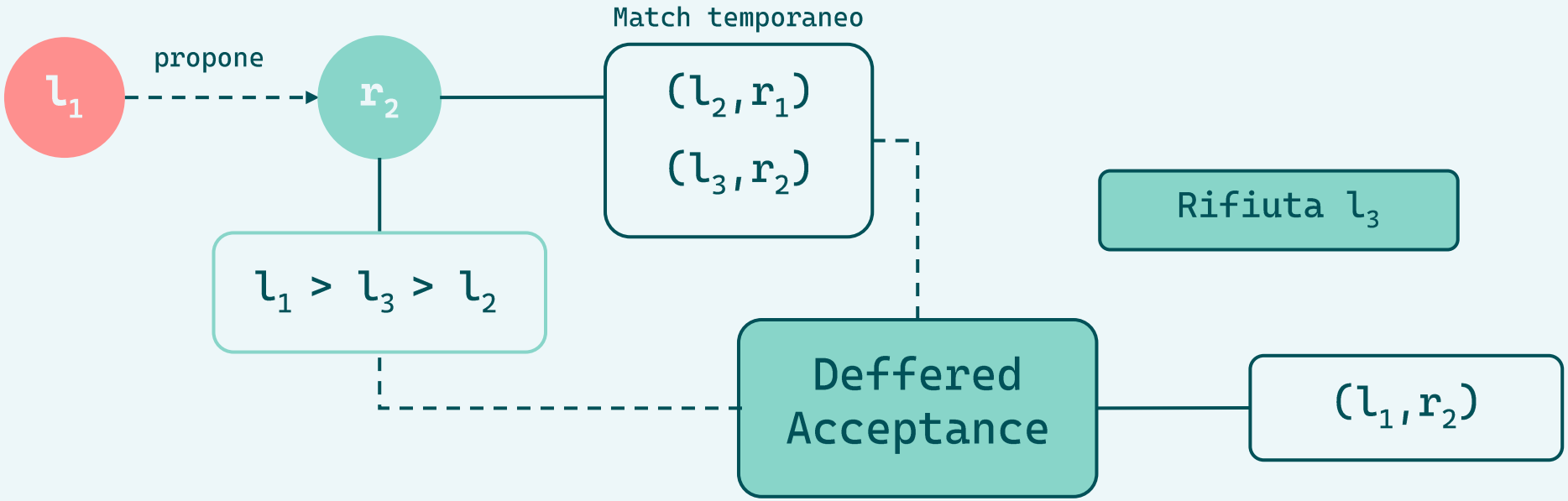
1° Round



A fine round unico senza match è l_1
l_1 aggiorna la sua lista di preferenza

l_1 → ~~r_1~~ $> r_2 > r_3$

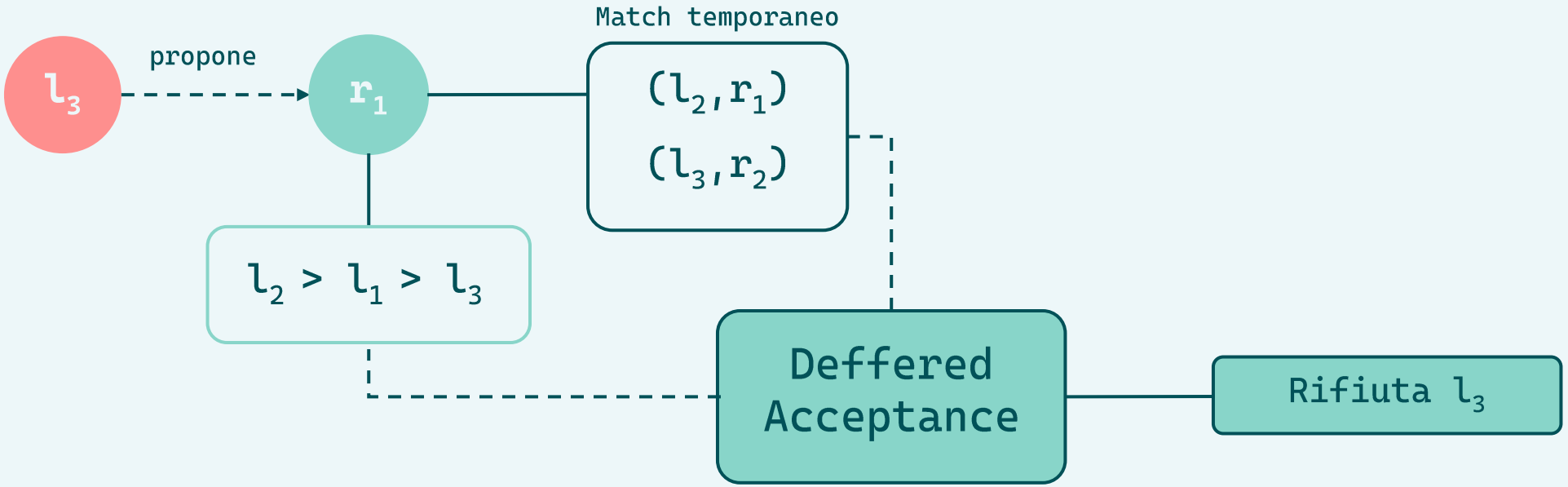
2° Round



A fine round unico senza match è l_3
l_3 aggiorna la sua lista di preferenza

l_3 → ~~r_2~~ $> r_1 > r_3$

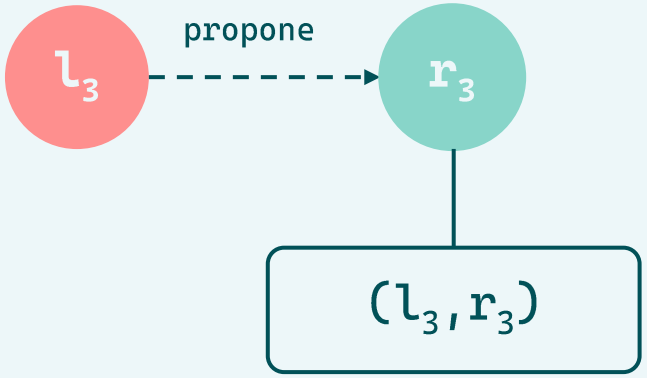
3° Round



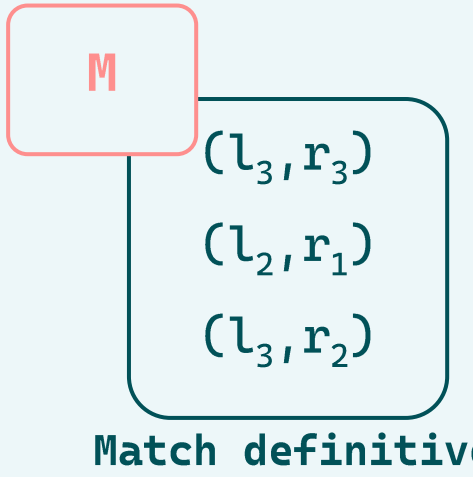
A fine round unico senza match è l_3
l_3 aggiorna la sua lista di preferenza

l_3 → ~~r_2~~ > ~~r_1~~ > r_3

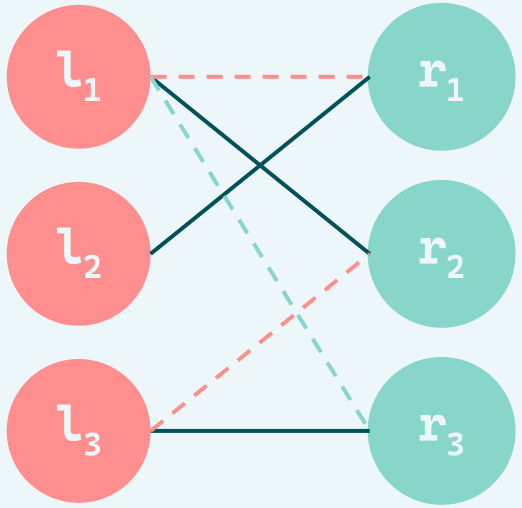
4° Round



Tutti hanno un match.
Il match ottenuto è stabile.



Match definitivo



Lo stable matching può essere raggiunto tramite **decisioni locali**

Il sistema **converge** verso la stabilità

L’algoritmo è **deterministico**

L $\{l_1 \dots l_k\}$

R $\{r_1 \dots r_k\}$

Ogni processore esegue un protocollo

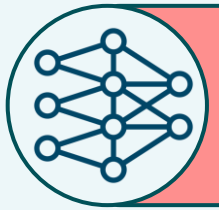
Processori distribuiti

La comunicazione si esegue tramite messaggi

Obiettivo:
Trovare un **matching stabile per un** sistema distribuito, dove ogni agente conosce solo la propria lista di preferenza

Sincronizzazione

Presenza di guasti



Struttura di rete

Siamo in un **modello sincrono** caratterizzato da un **orologio globale**

Tempo suddiviso in round discreti

Siamo in un **modello sincrono** caratterizzato da un
orologio globale

Tempo suddiviso in round discreti

I processi iniziano nello stesso momento

Siamo in un **modello sincrono** caratterizzato da un
orologio globale

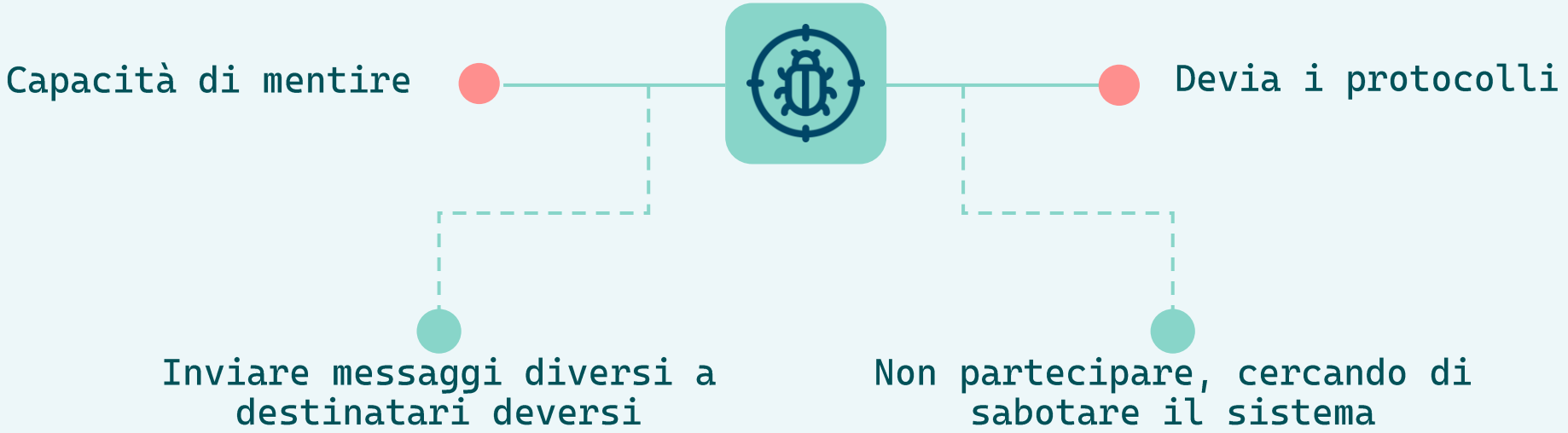
Tempo suddiviso in round discreti

I processi iniziano nello stesso momento

Limite massimo di consegna messaggi Δ

Obiettivo:

Trovare un **matching stabile per un sistema distribuito in presenza di agenti bizantini**

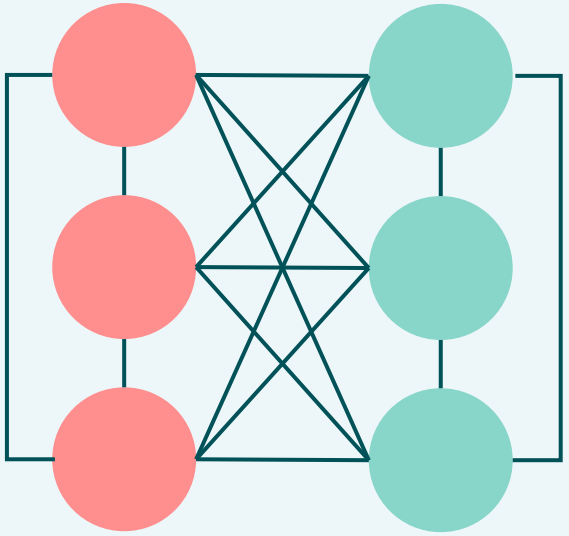


La possibilità di raggiungere uno stable matching, non dipende solo dal numero di agenti bizantini ma anche dalla **topologia di rete**

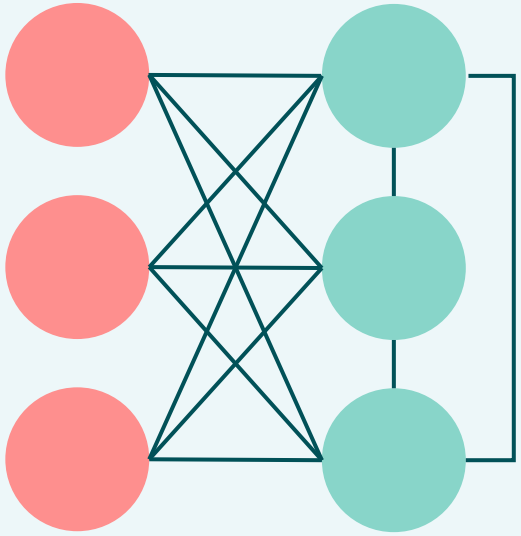
L $\{l_1 \dots l_k\}$

R $\{r_1 \dots r_k\}$

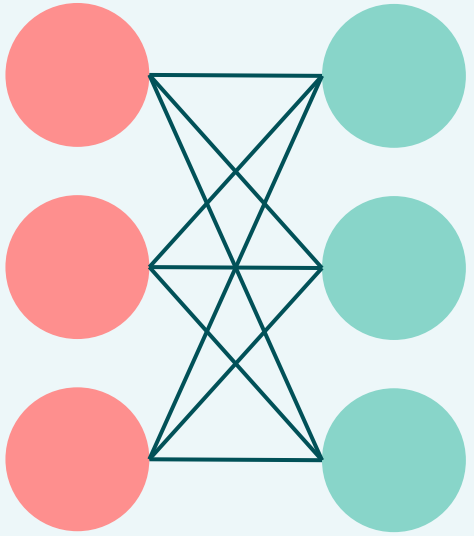
Fully connected



One sided (R)



Bipartite



La possibilità di raggiungere uno stable matching, non dipende solo dal numero di agenti bizantini ma anche dalla **topologia di rete**

Topologia	Senza firme	Con firme
Fully connected	< k/3 (almeno un lato)	< k (almeno un onesto per lato)
One-sided	< k/2 per lato	< k per lato
Bipartite	< k/2 per lato	< k per lato

la firma digitale è l'elemento che permette di **elevare la tolleranza ai guasti**

Per raggiungere uno stable matching in presenza di agenti bizantini dobbiamo garantire che gli onesti concordino sulle informazioni rilevanti

Necessitiamo di due protocolli:

Byzantine Broadcast

È un protocollo di accordo distribuito, tollerante ai fault bizantini

ogni agente usa il BB per "disseminare" la propria lista di preferenze

tutte le parti oneste ottengono una visione identica

Per raggiungere uno stable matching in presenza di agenti bizantini dobbiamo garantire che gli onesti concordino sulle informazioni rilevanti

Necessitiamo di due protocolli:

Byzantine

È un protocollo di accordo dist
bizantini

ogni agente usa il BB per "disseminare" la propria lista di preferenze

tutte le parti oneste ottengono una visione identica

Si trasforma il problema distribuito in una simulazione centralizzata

Stesso input -> Stesso matching

Per raggiungere uno stable matching in presenza di agenti bizantini dobbiamo garantire che gli onesti concordino sulle informazioni rilevanti

Necessitiamo di due protocolli:

Legame BSM-> BB

Posso perciò dire:
Per risolvere il problema BSM se è possibile risolvere il BB sulla topologia considerata e dalla presenza o meno di autenticazione

I limiti per il BSM riflettono esattamente i limiti del BB

Per raggiungere uno stable matching in presenza di agenti bizantini dobbiamo garantire che gli onesti concordino sulle informazioni rilevanti

Necessitiamo di due protocolli:

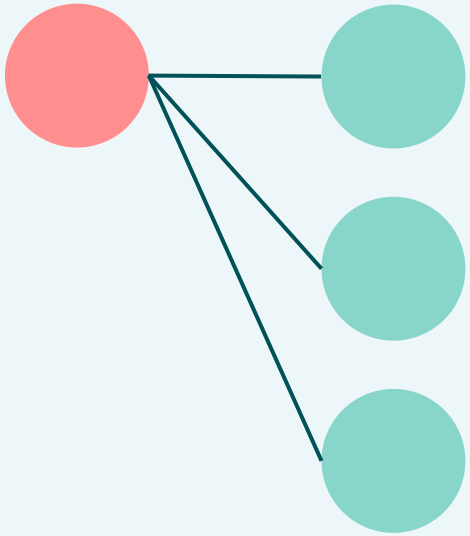
Byzantine Agreement

È un protocollo in cui le parti oneste devono raggiungere un consenso assoluto

tutti i decisori concordano su un valore

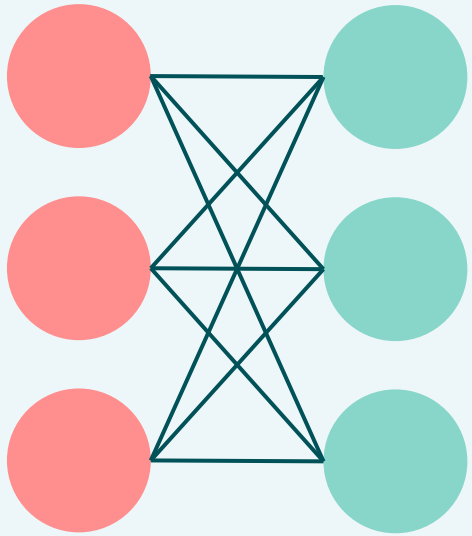
Per raggiungere uno stable matching in presenza di agenti bizantini dobbiamo garantire che gli onesti concordino sulle informazioni rilevanti

Necessitiamo di due protocolli:



BB

BA



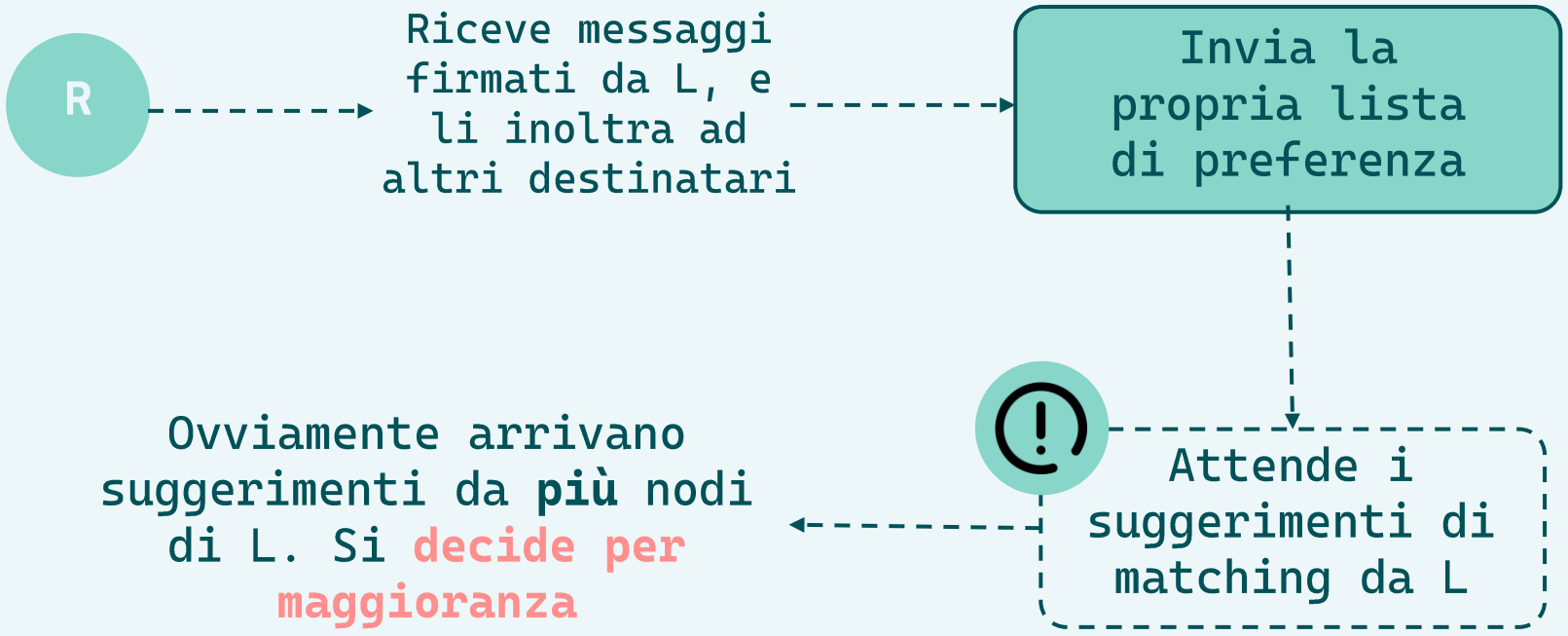
Parte $P \in R$

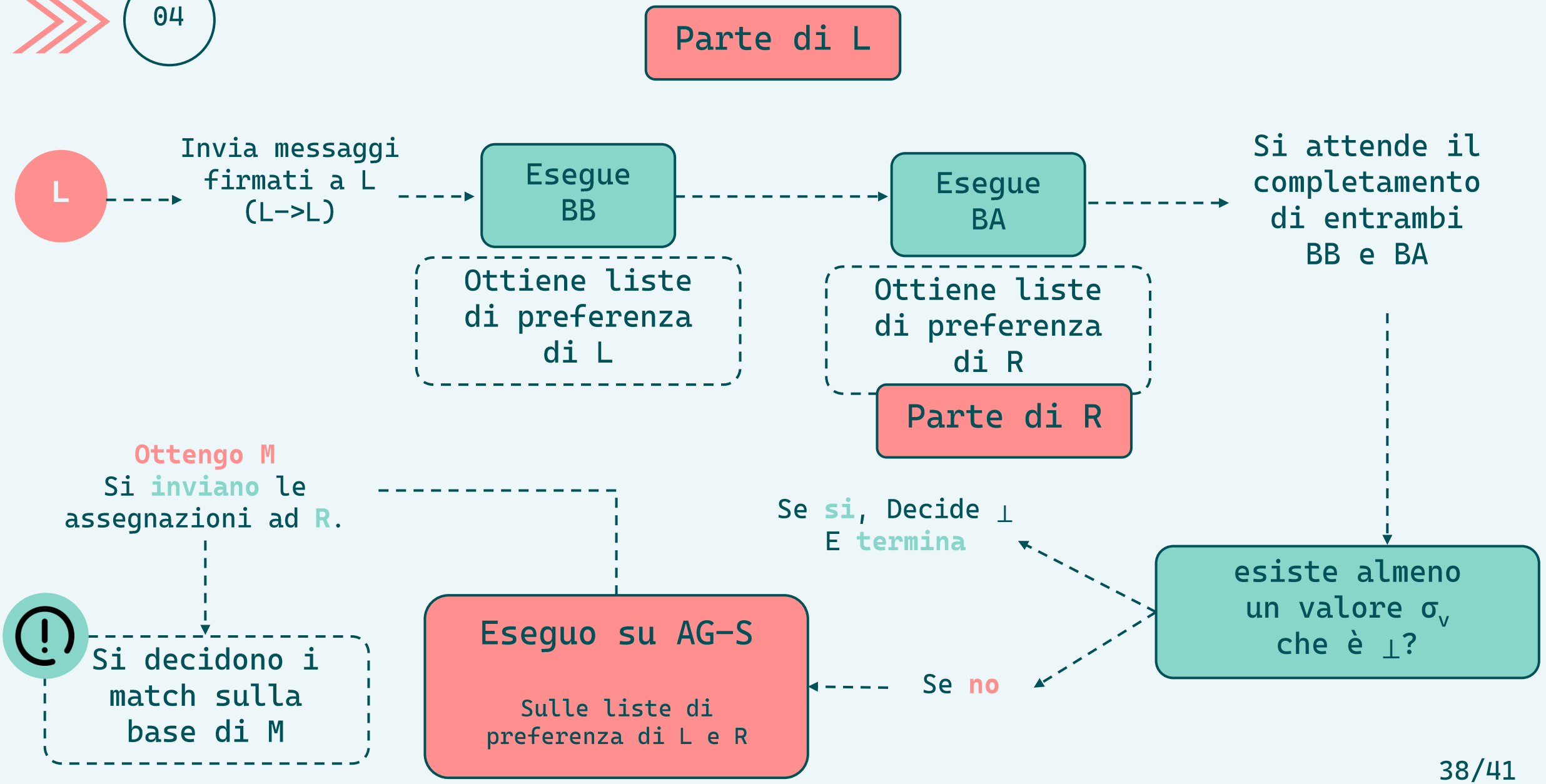
1. Inoltra messaggi firmati ricevuti da L.
2. Invia la propria lista a L.
3. Attendi suggerimenti di matching.
4. Decide per maggioranza tra i suggerimenti ricevuti.

Parte $P \in L$

1. Simulazione comunicazione L→L:
   Invia messaggi firmati tramite relay in R.
2. Raccolta preferenze:
   - Esegui Π_{BB} per ottenere $(\sigma_\ell)_{\{\ell \in L\}}$
   - Esegui Π_{BA} per ottenere $(\sigma_r)_{\{r \in R\}}$.
3. Sincronizzazione: Attesa completamento di Π_{BB} e Π_{BA}.
4. Se $\exists\ \sigma_v = \perp$:
   decide $\perp$ e termina.
5. Esegui AG-S su
   $((\sigma_\ell)_{\{\ell \in L\}}, (\sigma_r)_{\{r \in R\}})$
   ottenendo M.
6. Invia a R le assegnazioni e decide secondo M.

Parte di R


R
Riceve messaggi firmati da L, e li inoltra ad altri destinatari
Invia la propria lista di preferenza
Attende i suggerimenti di matching da L
Ovviamente arrivano suggerimenti da **più** nodi di L. Si decide per maggioranza


Parte di L
L
Invia messaggi firmati a L (L->L)
Esegue BB
Ottiene liste di preferenza di L
Esegue BA
Ottiene liste di preferenza di R
Parte di R
Si attende il completamento di entrambi BB e BA
esiste almeno un valore σ_v che è $\perp$?
Se si, Decide $\perp$ E termina
Se no
Eseguo su AG-S
Sulle liste di preferenza di L e R
Si decidono i match sulla base di M
Ottengo M
Si inviano le assegnazioni ad R.

Parte $P \in R$

1. Inoltra messaggi firmati ricevuti da L.
2. Invia la propria lista a L.
3. Attendi suggerimenti di matching.
4. Decide per maggioranza tra i suggerimenti ricevuti.

Parte $P \in L$

1. Simulazione comunicazione L→L:
   Invia messaggi firmati tramite relay in R.
2. Raccolta preferenze:
   - Esegui Π_{BB} per ottenere $(\sigma_\ell)_{\{\ell \in L\}}$
   - Esegui Π_{BA} per ottenere $(\sigma_r)_{\{r \in R\}}$.
3. Sincronizzazione: Attesa completamento di Π_{BB} e Π_{BA}.
4. Se $\exists\ \sigma_v = \perp$:
   decide $\perp$ e termina.
5. Esegui AG-S su
   $((\sigma_\ell)_{\{\ell \in L\}}, (\sigma_r)_{\{r \in R\}})$
   ottenendo M.
6. Invia a R le assegnazioni e decide secondo M.

Reference Bibliografiche

01

Gale, D., and L. S. Shapley. “College Admissions and the Stability of Marriage.” *The American Mathematical Monthly*, vol. 69, no. 1, 1962, pp. 9–15. *JSTOR*, https://doi.org/10.2307/2312726. Accessed 3 Mar. 2026.

02

Andrei Constantinescu, Marc Dufay, Diana Ghinea, and Roger Wattenhofer. 2025. Byzantine Stable Matching. In Proceedings of the ACM Symposium on Principles of Distributed Computing (PODC '25). Association for Computing Machinery, New York, NY, USA, 181–191. https://doi.org/10.1145/3732772.3733525

03

https://web.stanford.edu/class/msande319/MatchingSpring19/lecture08.pdf

04

Grazie per l’attenzione

Montella Geraldine
Università degli studi di Salerno
Mentore: Roberto De Prisco