

Implementazione del protocollo/algoritmo di Chord

*Corso di SISTEMI DISTRIBUITI E CLOUD COMPUTING
Facoltà di INGEGNERIA INFORMATICA*



Simone Festa, mat. 0320408

Agenda

Introduzione

Server
Registry

Finger Table

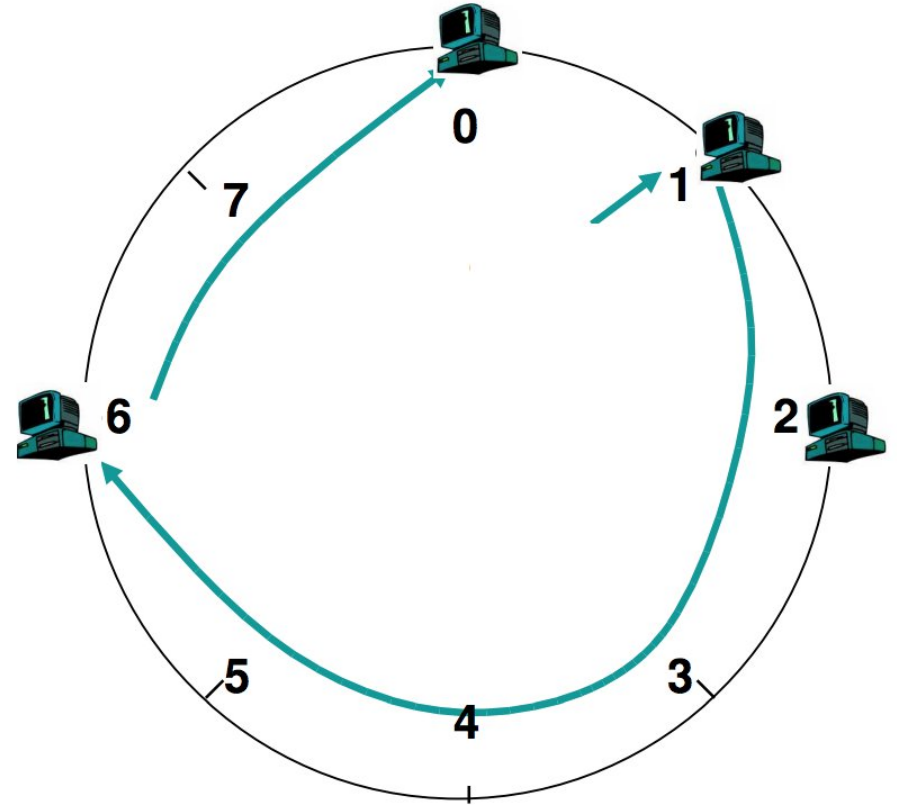
Gestione
risorse

Join/Leave

Conclusioni

Introduzione

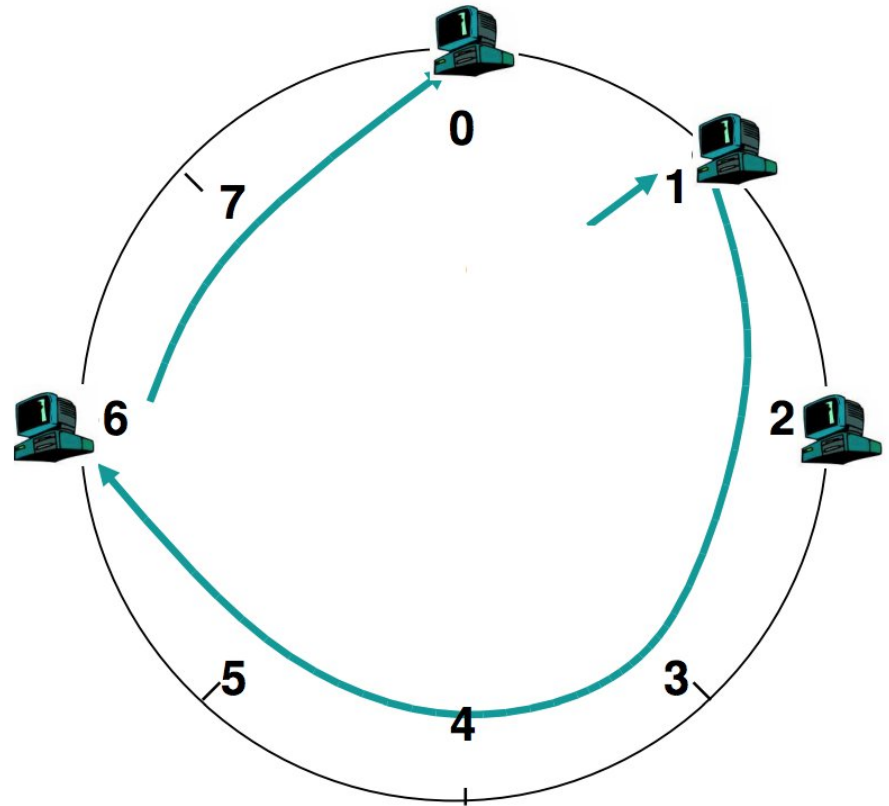
- Lo scopo del progetto è implementare il protocollo/algorithmo di **Chord**.
- Di seguito, viene proposta una rappresentazione del sistema su cui Chord si basa:
- Disposizione ad *anello* a livello *overlay network strutturato*.



Introduzione

- Nodi e Risorse vengono mappati nello stesso spazio contiguo (**consistent hashing**)
- Come si introduce un nuovo nodo nella rete?
- Come comunicano i nodi nella rete?
- Come vengono assegnate le risorse?

Chord ha l'obiettivo di definire tali modalità!



Agenda



Server Registry

- IP statico e noto a tutti.
Permette ad un *client esterno* di interagire col sistema.
- Fornisce ad un *nuovo* nodo un punto di accesso nel sistema, mantenendo la lista dei nodi presenti.
- Fornisce supporto per il controllo di eventuali nodi caduti.




Agenda




Finger Table

- Ogni nodo possiede una propria Finger Table. Tutte le FT hanno lo stesso numero di righe, pari ad “ m ”, cioè il numero di bit usati per un identificativo.
- La riga i -esima della FT di un *nodo* p , è così calcolata:

$$FT_p[i] = \text{succ}(p + 2^{i-1}) \bmod 2^m$$

 *Nodo responsabile dell'identificativo posto come argomento.
Tale operazione è svolta da un nodo presente nel sistema.*



Finger Table

i	$id + 2^i$	succ
0	2	2
1	3	6
2	5	6

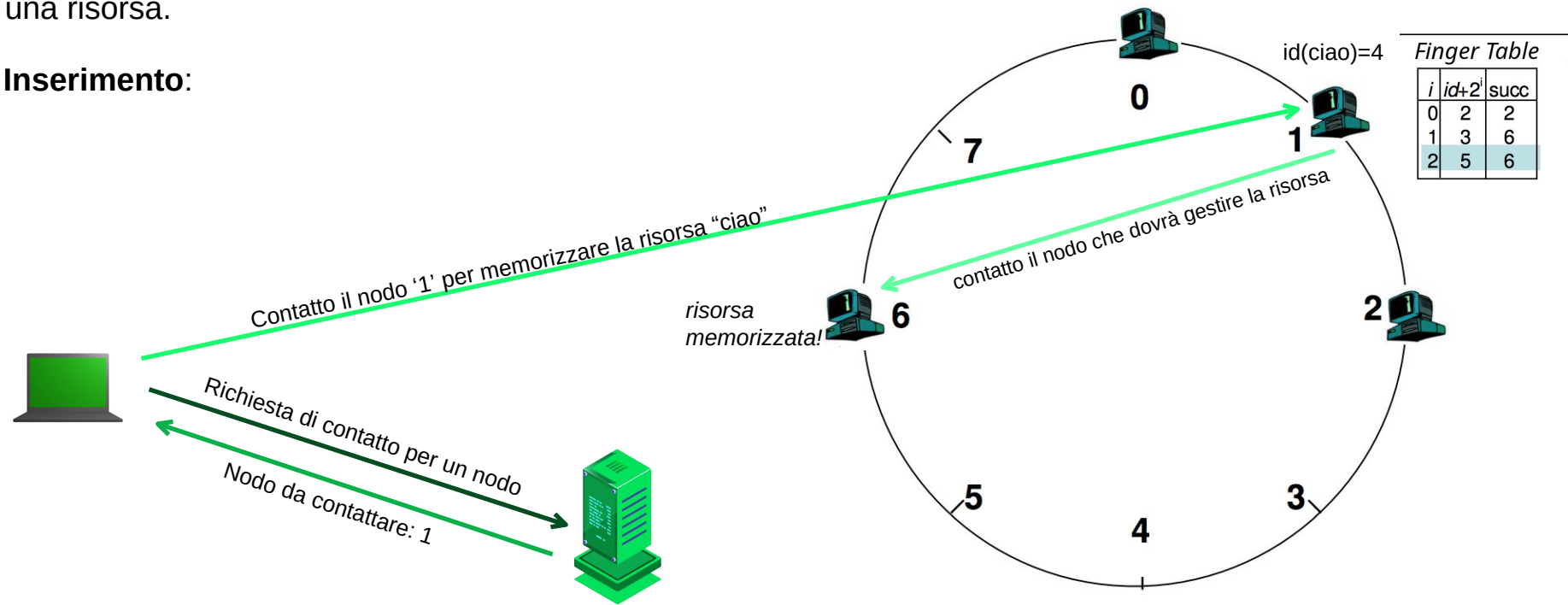
- La Finger Table mantiene una lista di nodi progressivamente distanti. Fornisce una conoscenza ben definita dei nodi vicini e più approssimata all'aumentare della distanza.
- Ciò consente una ricerca veloce, in $O(\log N)$, senza interrogare tutto l'anello.

Agenda



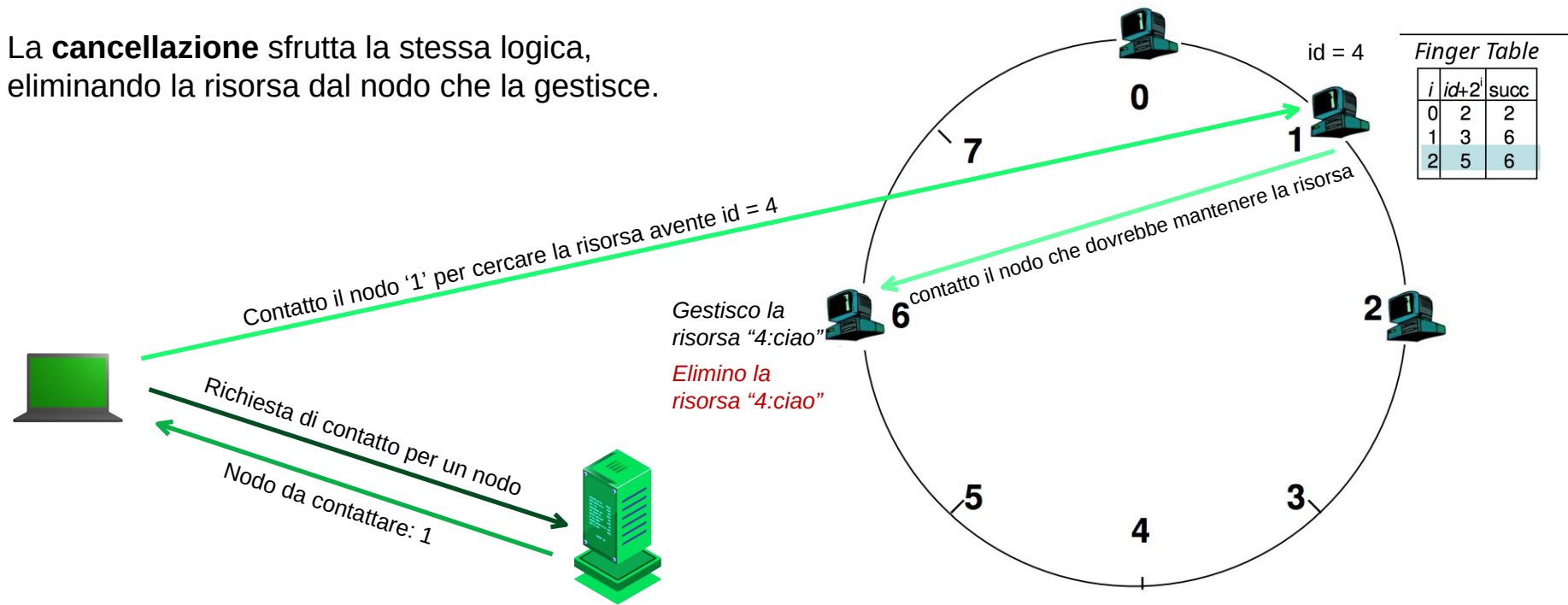
Gestione delle risorse

- Il sistema supporta l'*inserimento*, la *ricerca* e la *cancellazione* di una risorsa.
- Inserimento:**



Gestione delle risorse

- La **ricerca** di una risorsa è eseguita in modo simile.
- La **cancellazione** sfrutta la stessa logica, eliminando la risorsa dal nodo che la gestisce.

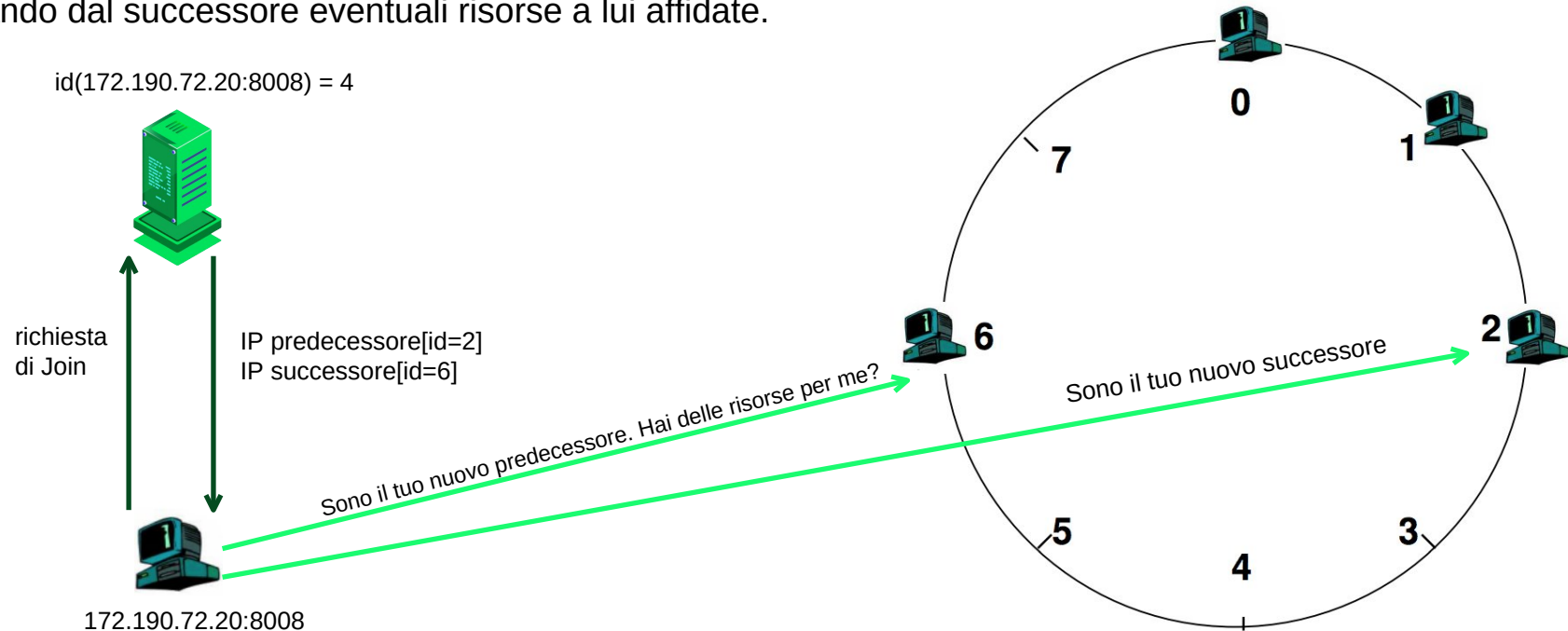


Agenda



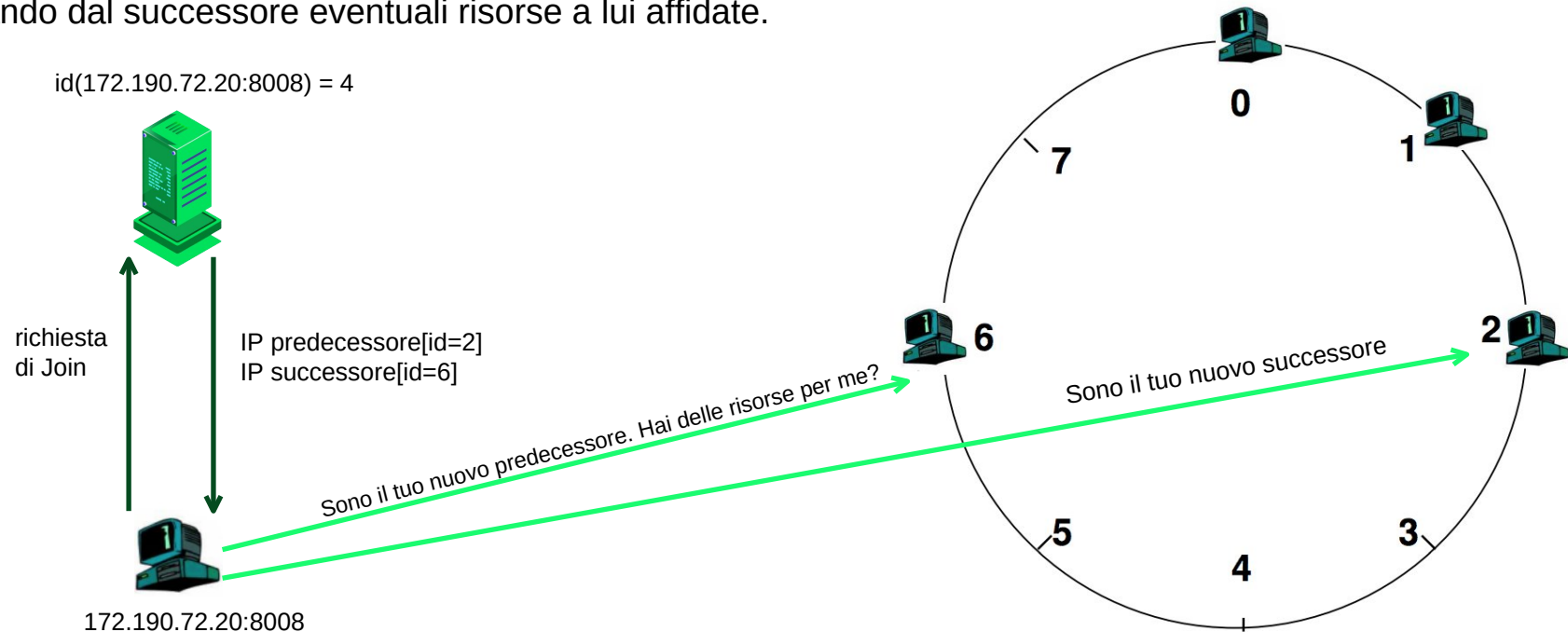
Join/Leave

- Nella fase di **Join**, il nodo entrante comunica col Registry per conoscere i suoi vicini. Li contatterà per inserirsi nel sistema, prelevando dal successore eventuali risorse a lui affidate.



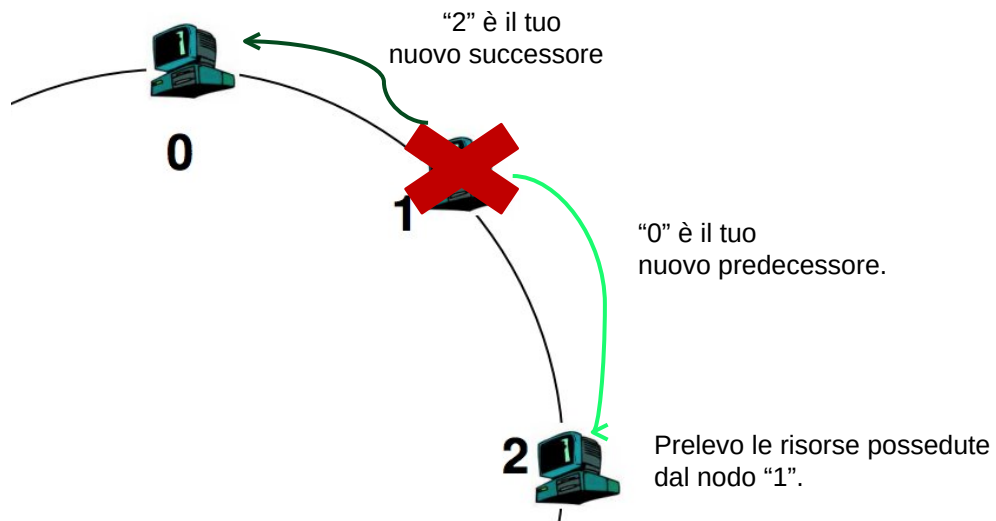
Join/Leave

- Nella fase di **Join**, il nodo entrante comunica col Registry per conoscere i suoi vicini. Li contatterà per inserirsi nel sistema, prelevando dal successore eventuali risorse a lui affidate.



Join/Leave

- Nella fase di **Leave** (controllata), i nodi adiacenti al nodo da rimuovere verranno contattati per aggiornare la loro conoscenza sui nodi predecessori e successori.
- Esempio: rimozione del nodo "1".



Agenda



Conclusioni

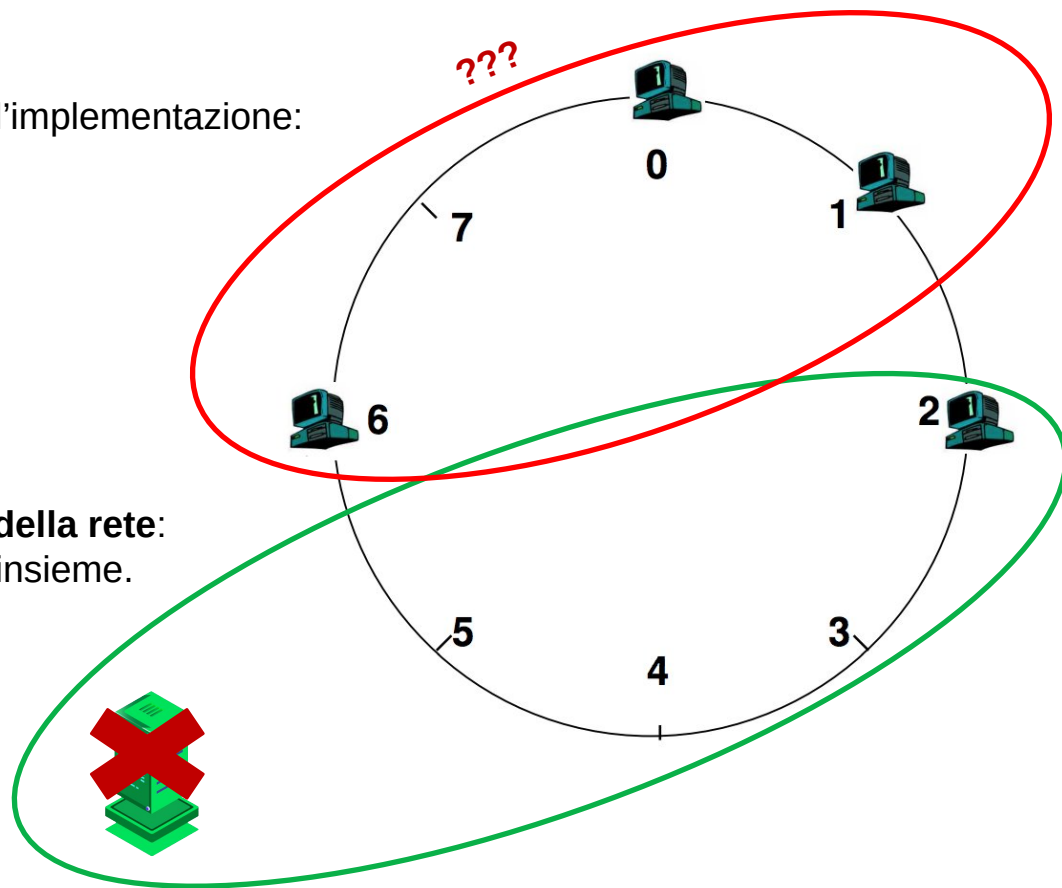
- Di seguito, esaminiamo alcune limitazioni dell'implementazione:

- **Server Registry bottleneck:**

un suo guasto limiterebbe alcune funzionalità per la gestione del sistema.

- **Non tolleranza in caso di partizionamento della rete:**

Il registry verrebbe associato ad un solo sottoinsieme.



Repository



GitHub

https://github.com/simonefesta/Chord_SDCC

Grazie per l'attenzione!

Simone Festa, mat. 0320408

