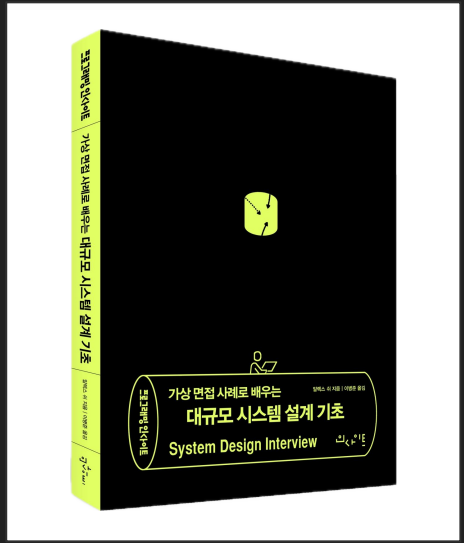
5장. 안정 해시 설계



1) 일반 해시

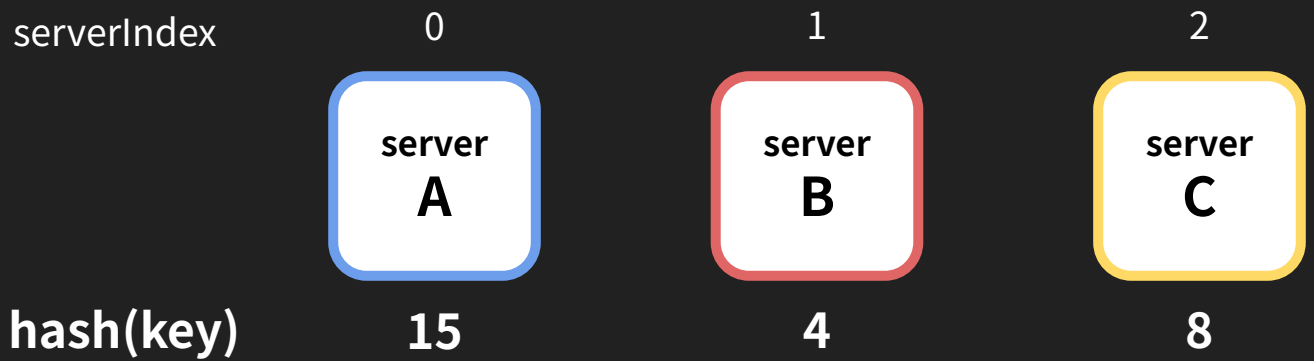
serverIndex = hash(key) % N

*N : 서버의 개수
*key : 요청 분배의 기준이 될 구분 값(e.g. userId)

● Hash 함수를 쓰는 이유? O(1), 저장공간 절약, 검색속도 향상!

1) 일반 해시

serverIndex = hash(key) % N



적절한 경우

- Server Pool 크기 고정
- 데이터 분포 균등

1) 일반 해시 - 단점

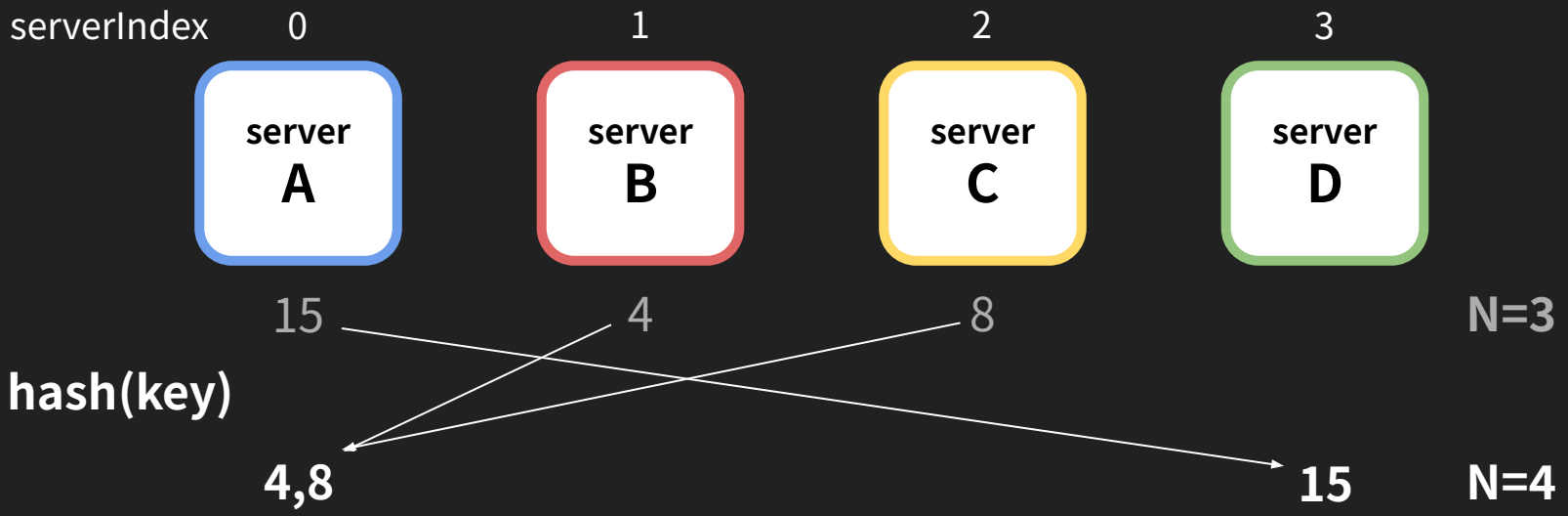
모듈러 연산 : 재분배되는 키의 수 ↑

저장된 캐시를 사용하지 못하게 되는 경우의 수 ↑

대규모 Cache Miss

1) 일반 해시 - 서버 추가

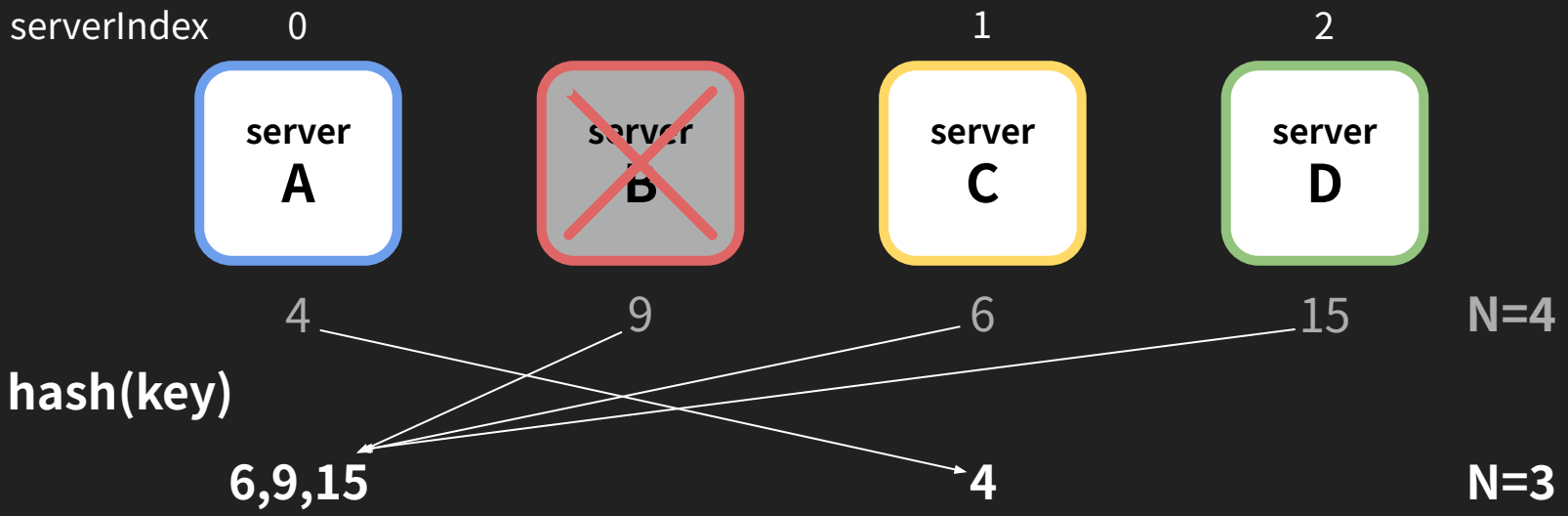
serverindex = hash(key) % N



대부분의 키 재분배!

1) 일반 해시 - 서버 삭제

serverindex = hash(key) % N



대부분의 키 재분배!

재분배되는 키의 수를 최대한 줄일 수 없을까?

2) 안정 해시 : Consistent Hash

위키백과

우리 모두의 백과사전

일관된 해싱

- 웹 서버의 개수 변동 시 -> 요청을 분산하는 방법
- 노드나 슬롯의 개수가 바뀔 때, 평균적으로 k/n의 키만 재매핑

모듈러 연산은 사용금지,

저장공간을 원형으로 만들고 해시값을 상대적 위치로 비교하자!

2) 안정 해시 : Consistent Hash

재배치 되는 키의 수 = k/n 개

*k : 전체 키의 수 / n : slot 수(노드의 수)

hash(serverKey) ≳ hash(key)

*≳ : 큰 수 중 가장 유사한 값

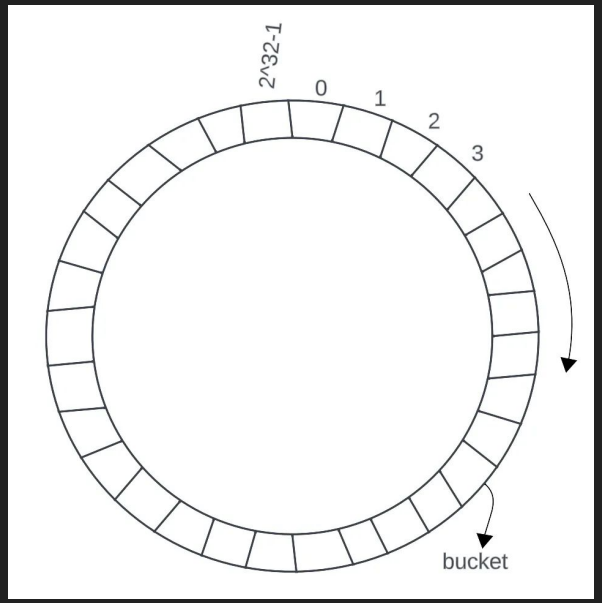
2) 안정 해시 : Consistent Hash

해시 함수 : f(x) (e.g. SHA-1)
함수의 출력값 : x0 ~ xn

- 오른쪽에 있을 수록 큰 수

- *큰 수 중 가장 유사한 값
= 오른쪽으로 탐색했을 때 가장 먼저 만나는 값

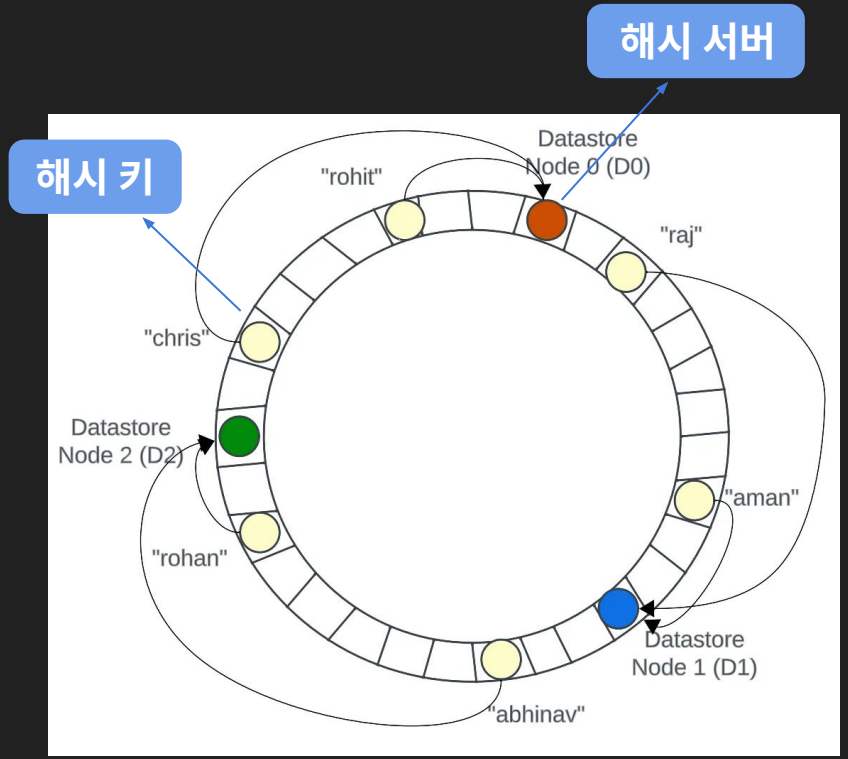
<해시 공간>



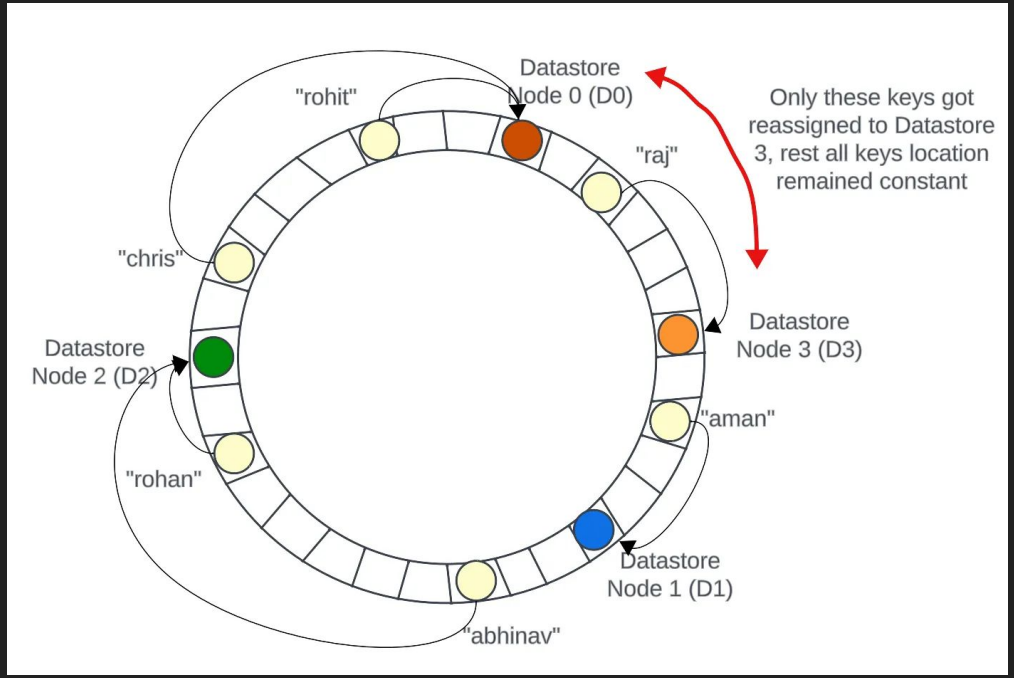
<해시 링>

2) 안정 해시 - 기본 구현

serverIndex ≳ hash(key)
***≳ : 큰 수 중 가장 유사한 값**

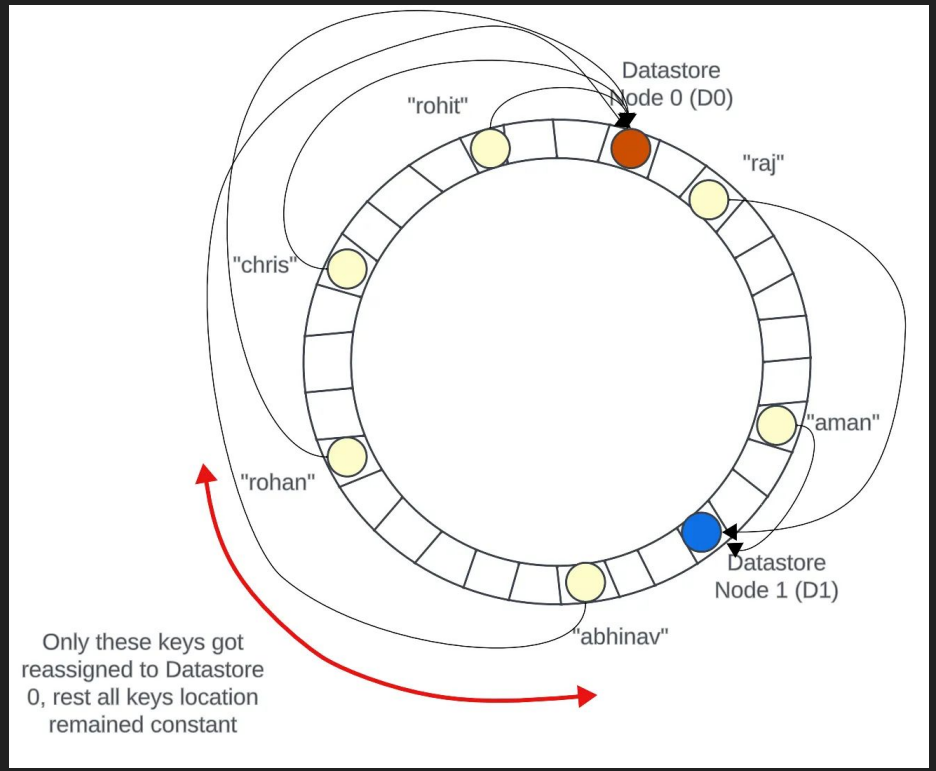


2) 안정 해시 - 서버 추가



일부 키만 재분배!

2) 안정 해시 - 서버 삭제

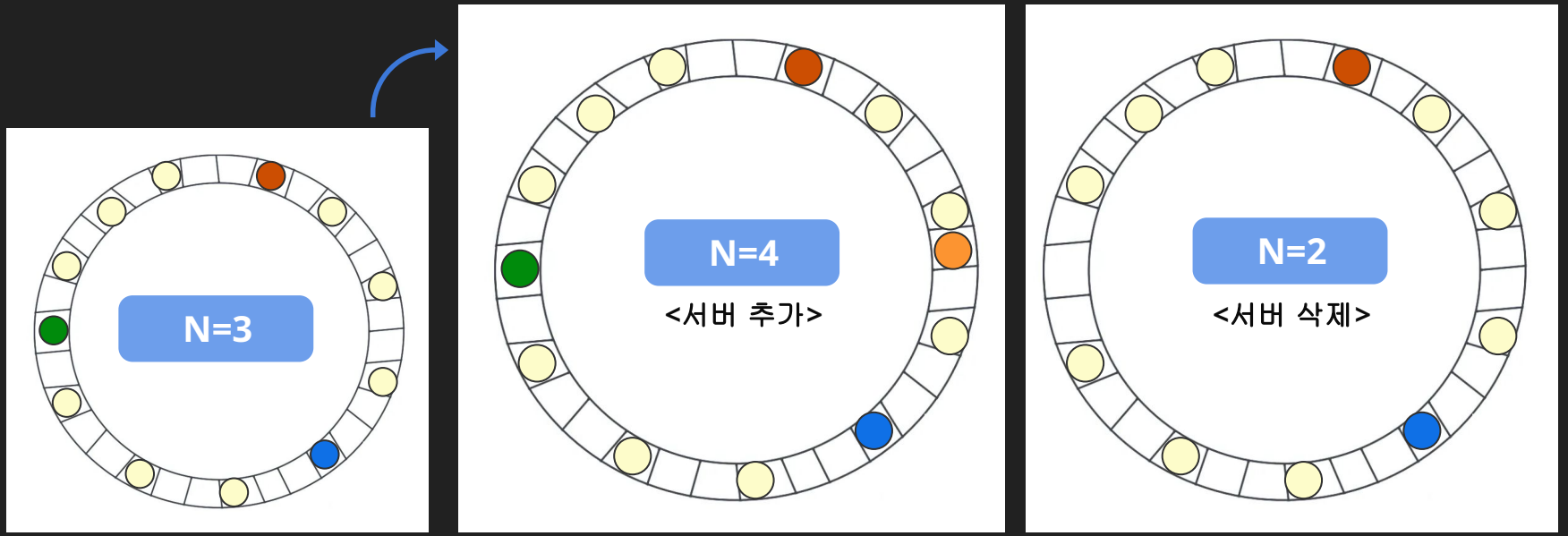


일부 키만 재분배!

2) 안정 해시 - 기본구현의 단점

① Partition 크기 균등하게 유지 X

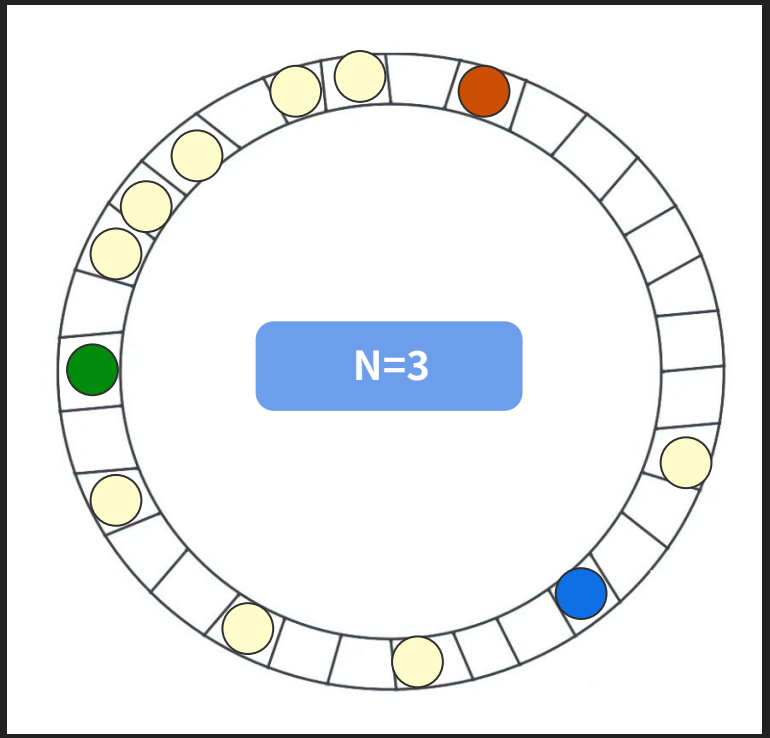
- 가정 : key 균등하게 분포



2) 안정 해시 - 기본구현의 단점

② 키 균등하게 분포 X

- 가정 : 서버 균등하게 분포
(uniform distribution)



2) 안정 해시 - 기본구현의 단점

① Partition 크기 균등하게 유지 X

② 키 균등하게 분포 X

특정 서버 부하

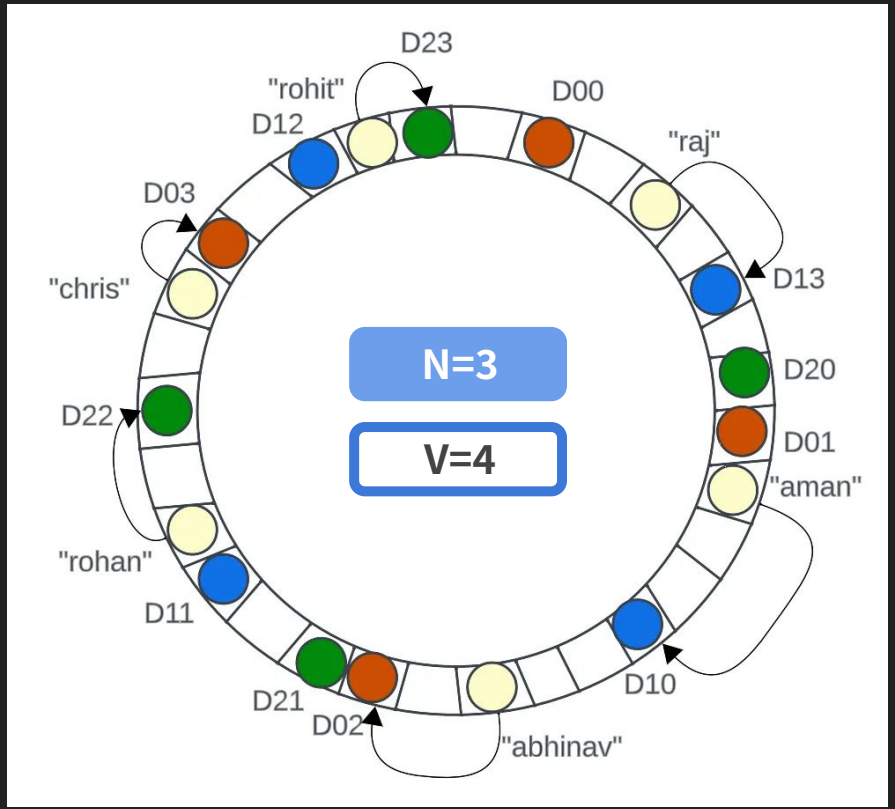
2) 안정 해시 - 가상 노드 : Virtual Node

- 복제(replica)

- e.g. V=3인 경우

- **A(0) ⇒ A(0)_0, A(0)_1, A(0)_2**
- **B(0) ⇒ B(0)_0, B(0)_1, B(0)_2**
- **C(0) ⇒ C(0)_0, C(0)_1, C(0)_2**

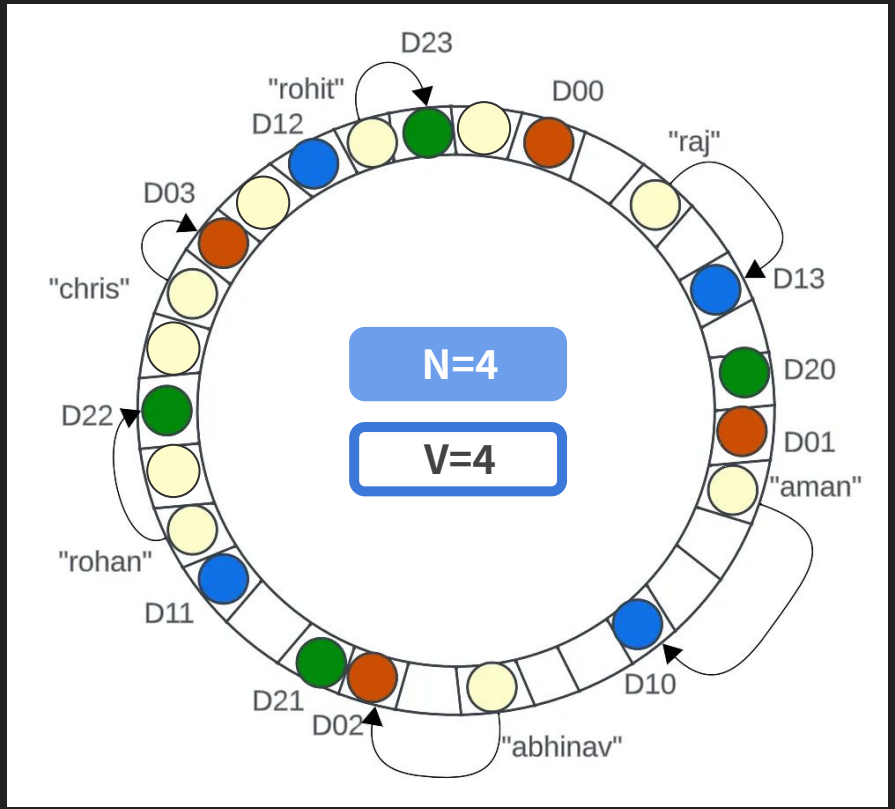
***server(index)_(replica number)**
***V : 가상 노드의 수**



2) 안정 해시 - 가상 노드의 장점

key가 특정 구간에 몰려도

여러 서버에 분배!



2) 안정 해시 - 가상 노드의 장점

가상노드의 개수 ↑

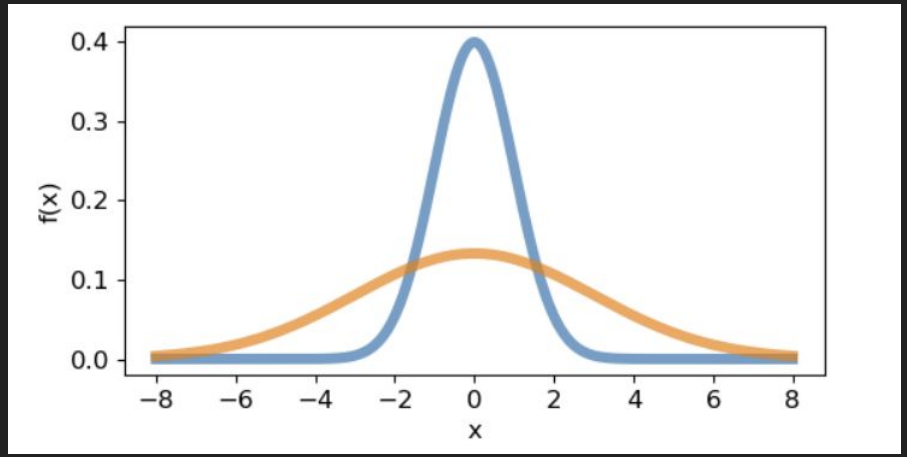
(e.g. 100 ≤ V ≤ 200, *V : 가상 노드의 수)

❯

키의 분포 균등

: 표준 편차(standard deviation) ↓

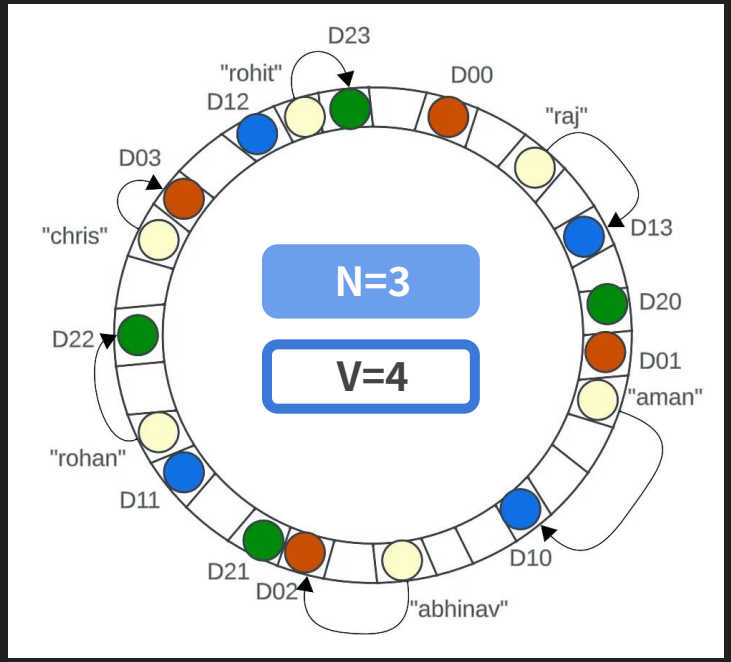
(e.g. 10% ≤ 표준편차 ≤ 5%)

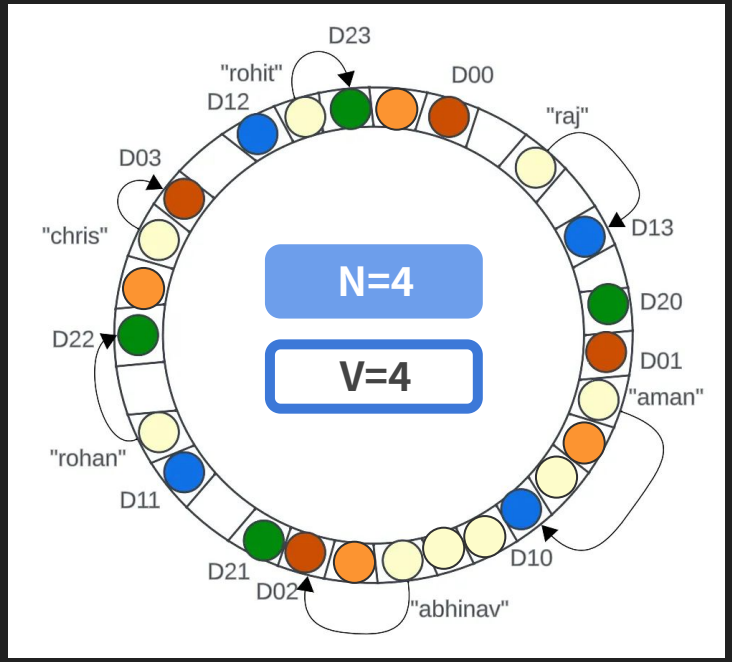


<각 서버가 담당하는 키의 개수 분포>

2) 안정 해시 - 가상 노드의 장점

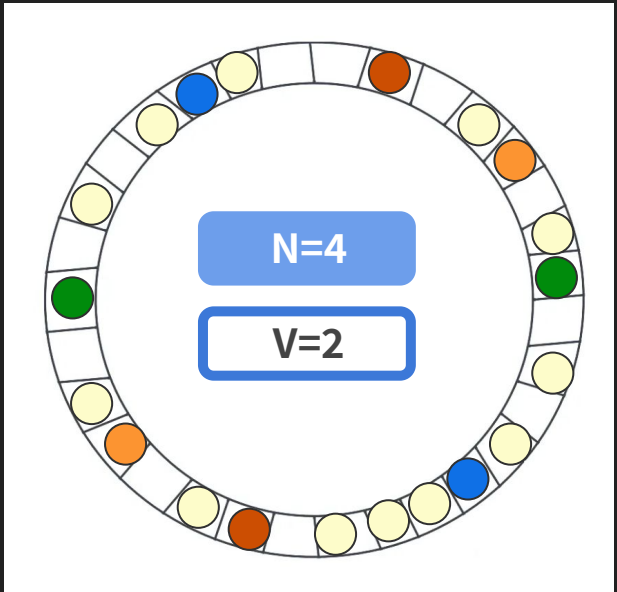
① Partition 크기 균등하게 유지 O

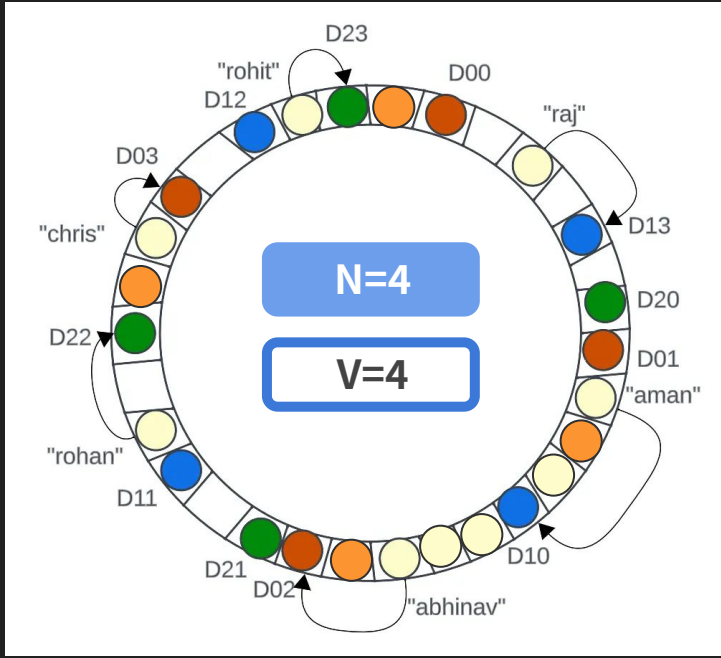




2) 안정 해시 - 가상 노드의 장점

② 키 균등하게 분포 O





2) 안정 해시 - 가상 노드 수의 결정

3) 참고문헌

- 가상면접 사례로 배우는 대규모 시스템 설계 기초
- https://vivekbansal.substack.com/p/what-is-consistent-hashing
- https://ko.wikipedia.org/wiki/일관된_해싱