알파고의 구성

알파고의 능력은 어디에서 오는가?

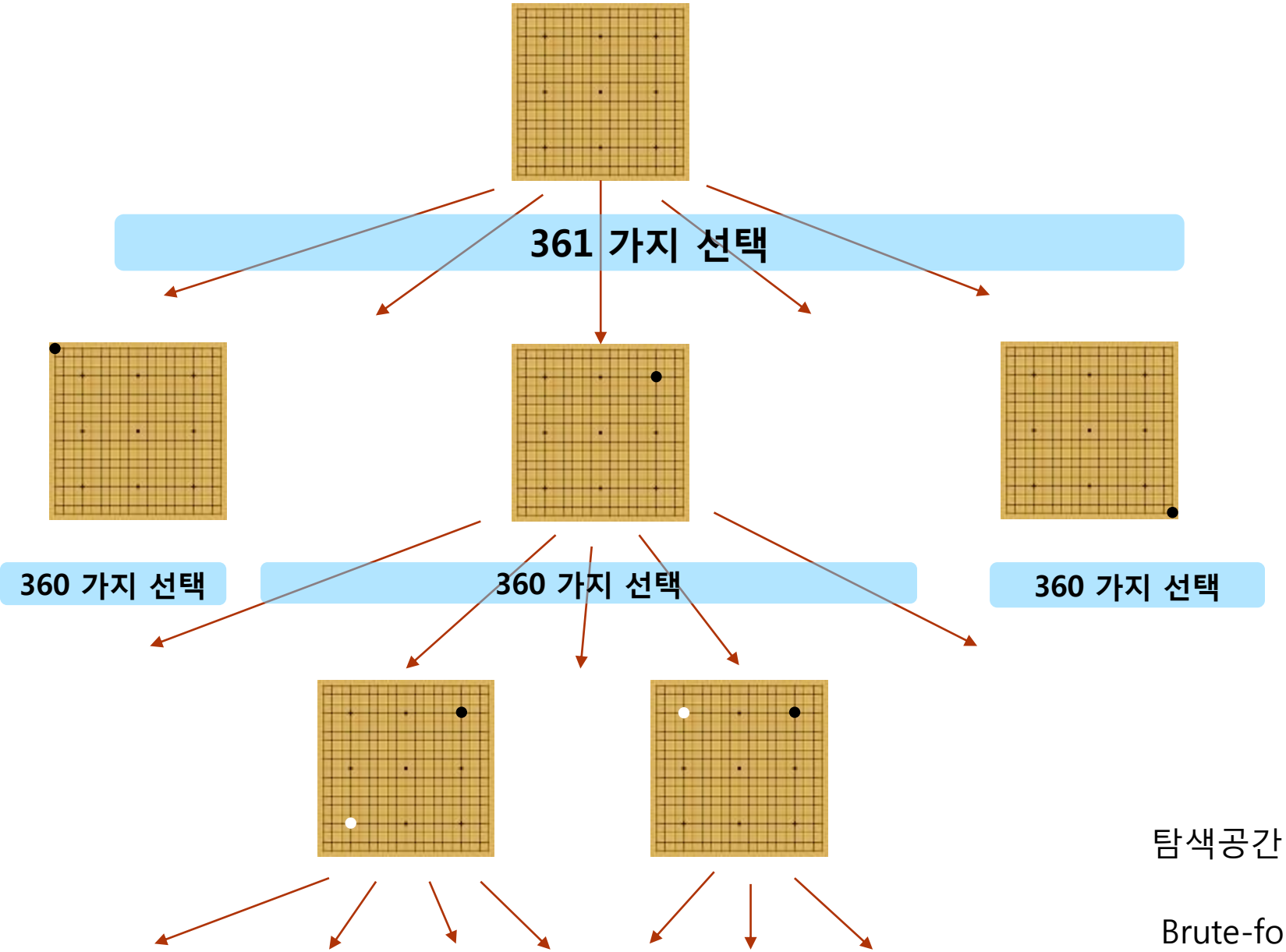
앱센터

2016. 03. 24

소프트웨어정책연구소
김석원 책임연구원
skimaza@spri.kr

SPRi 소프트웨어정책연구소

바둑 문제의 게임트리 표현



게임상태

착수(액션)

게임상태

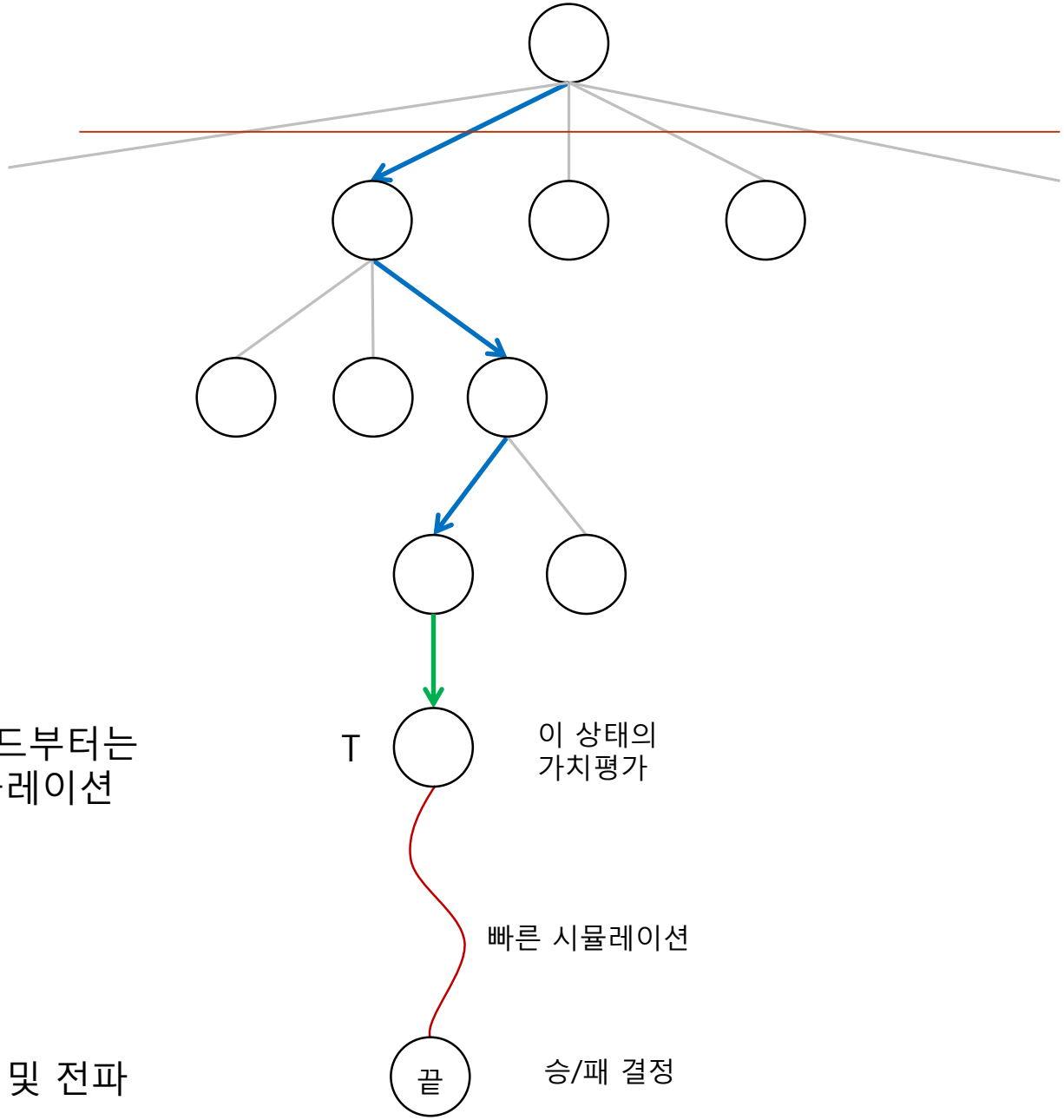
착수(액션)

게임상태

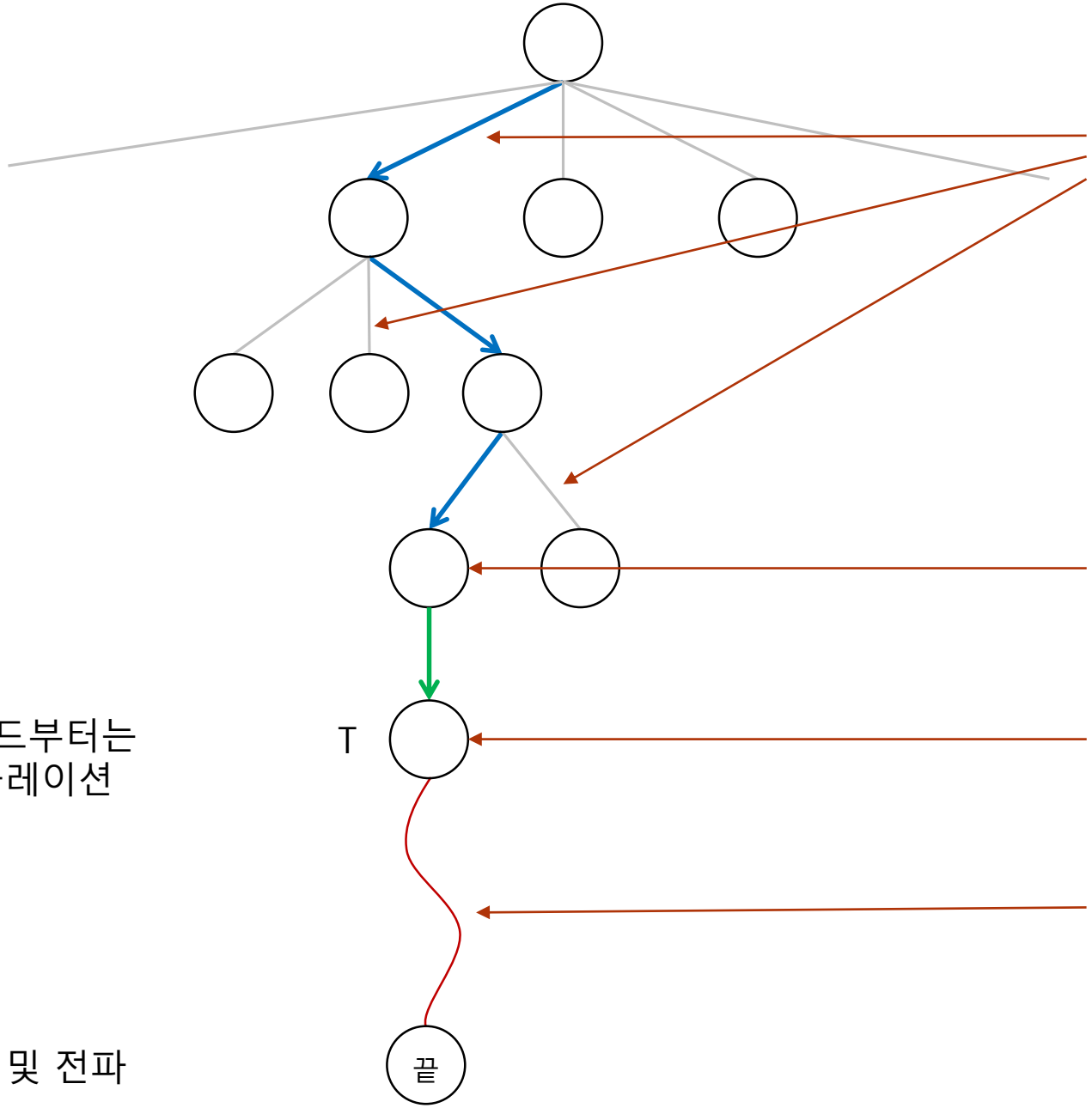
탐색공간 규모: $250^{150} \fallingdotseq 10^{360}$

Brute-force 방식은 부적절

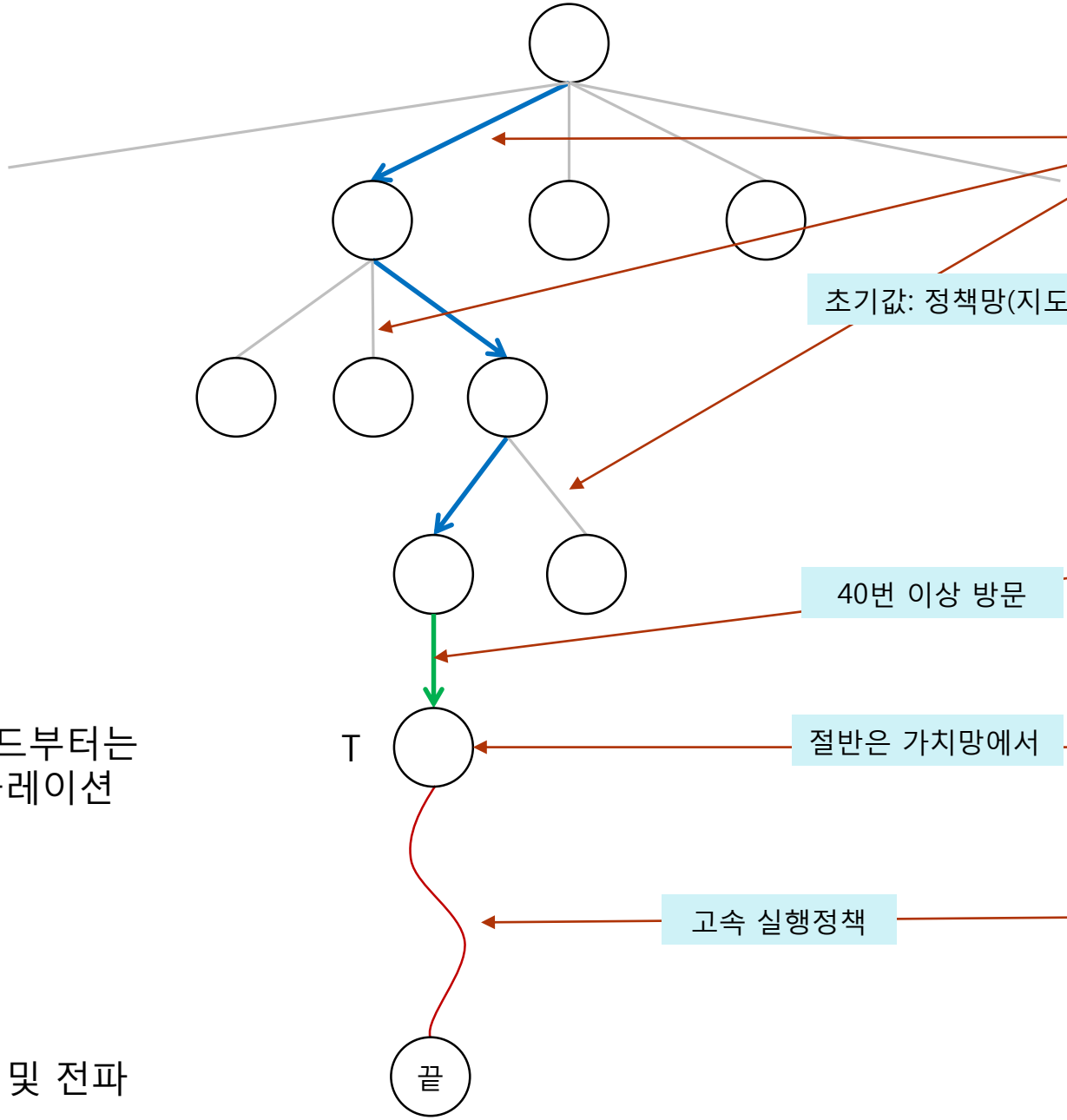
몬테카를로 트리탐색 – Coulom, 2006, Crazy Stone



MCTS를 하려면 무엇이 필요할까?



MCTS를 하려면 무엇이 필요할까? – 알파고의 방법



가장 많이 방문한 링크

무엇을 선택?

시뮬레이션 결과 반영

링크값 변경

언제 노드를 확장할까?

새로 펼쳐진 노드의 가치 평가

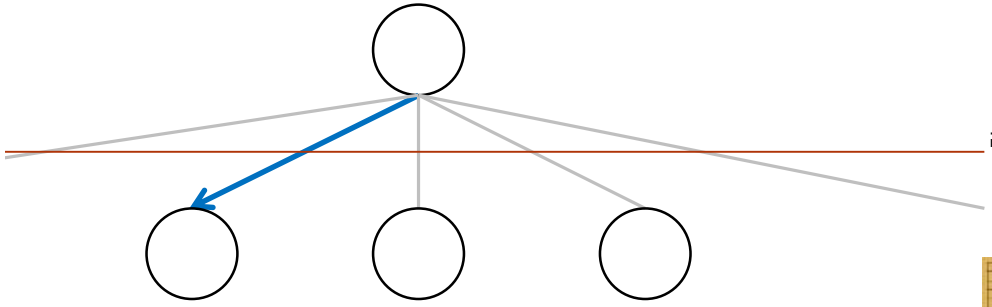
절반은 시뮬레이션 결과에서

빠른 시뮬레이션 방법

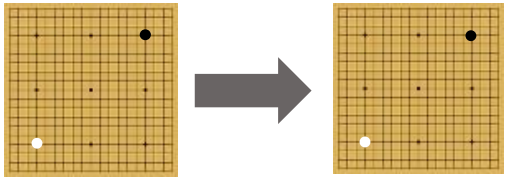
Backup

액션선택 – 링크의 초기값

Q + u
Q: 축적된 시뮬레이션 결과
u: 상태값. 시뮬레이션이 진행되며 점점 작아짐



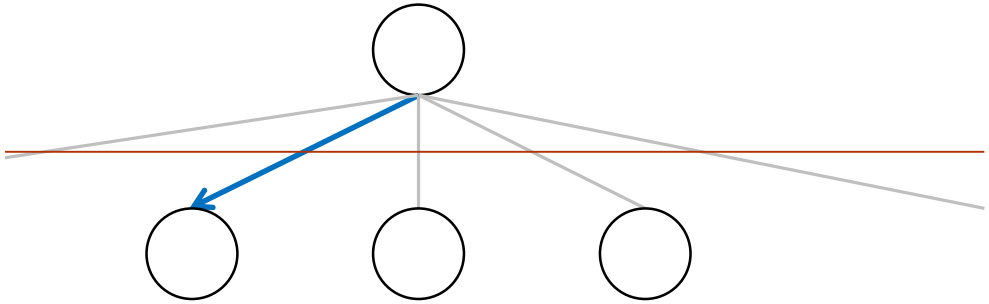
링크의 초기값을 지도학습 정책망으로 지정



다음 수의 위치별 확률분포

액션선택 – 엣지의 초기값

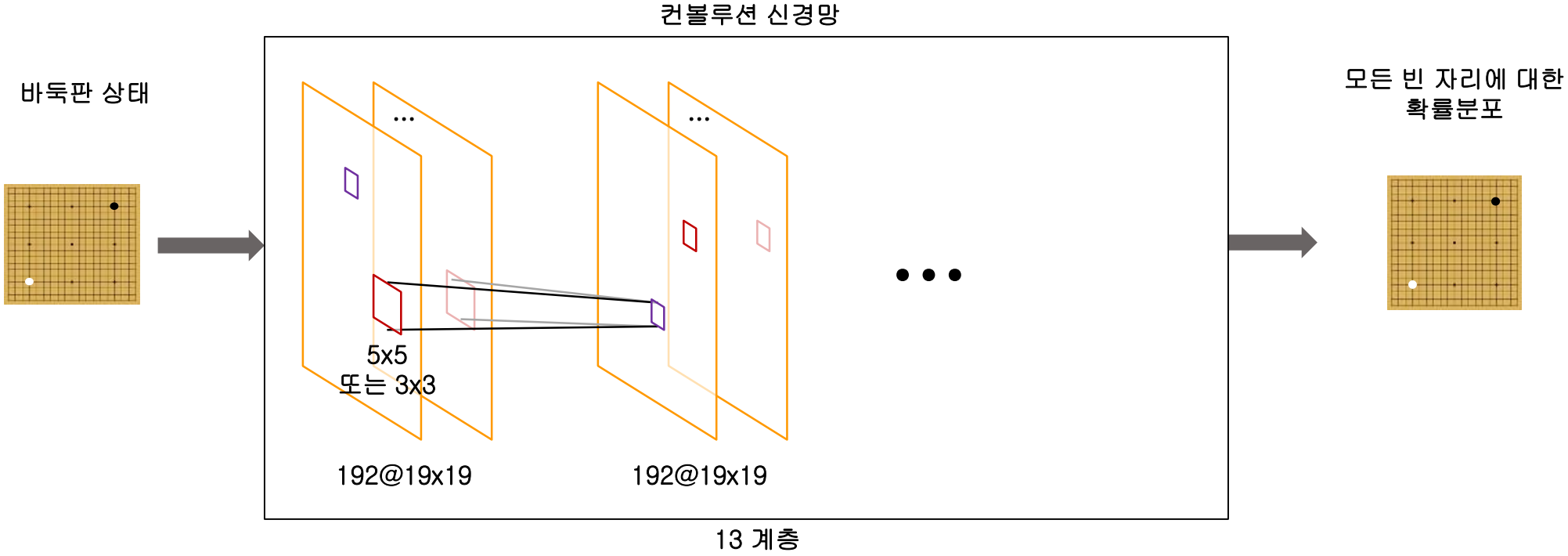
Q + u
Q: 축적된 시뮬레이션 결과
u: 상태값. 시뮬레이션이 진행되며 점점 작아짐



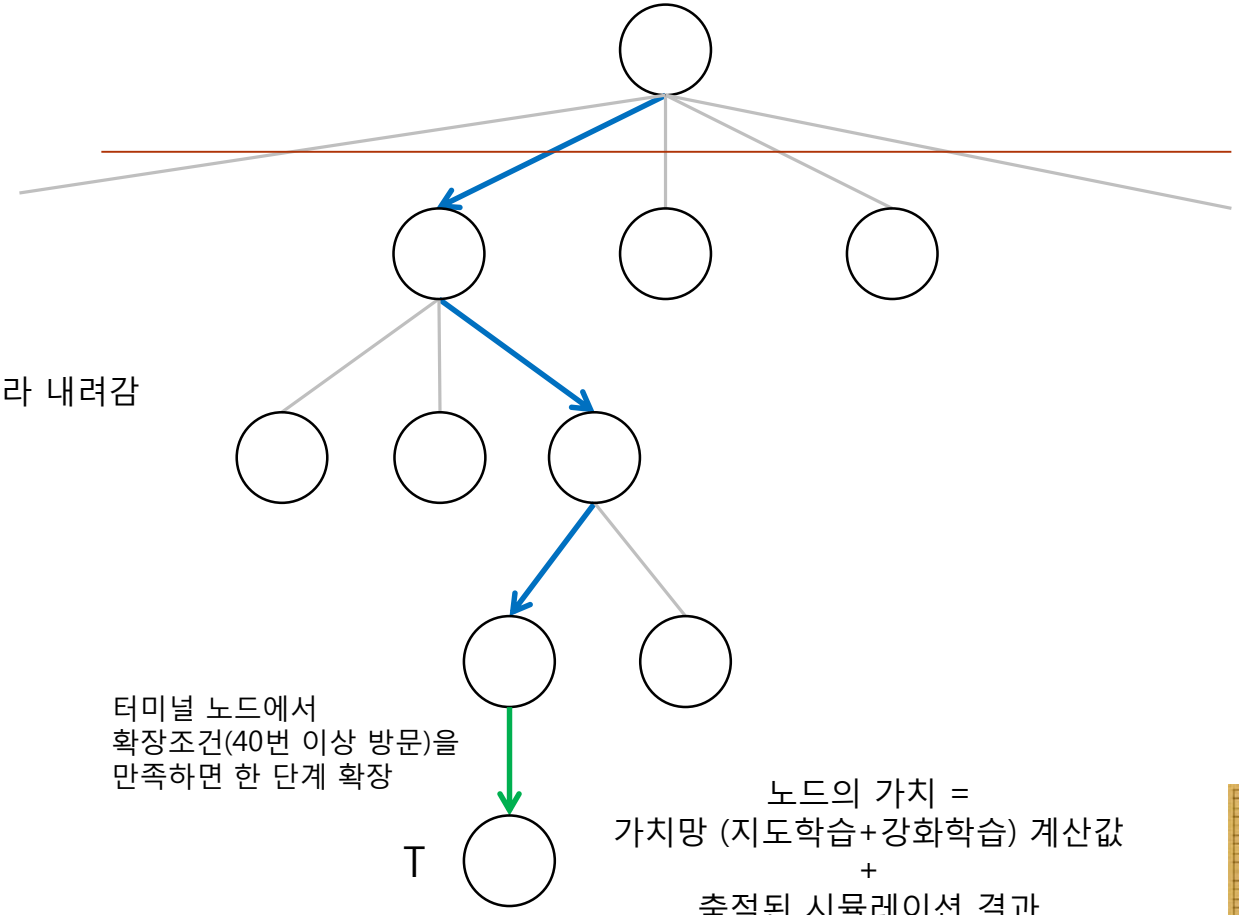
링크의 초기값을 지도학습 정책망으로 지정

어떤 상태에서 다음 수를 어디에 두는지에 대한 확률 분포를 학습

입력데이터
KGS 6-9단의
16만 게임
3천만가지 착수



노드의 가치 계산 1 – 상태를 입력으로 가치망에서 계산

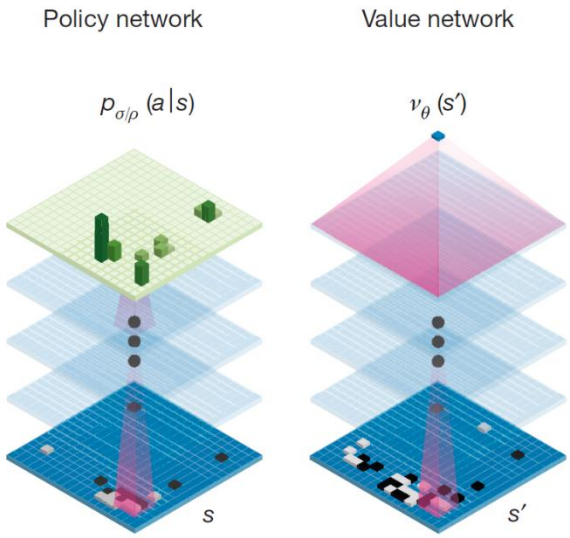


트리탐색
최대값을 따라 내려감

터미널 노드에서
확장조건(40번 이상 방문)을
만족하면 한 단계 확장

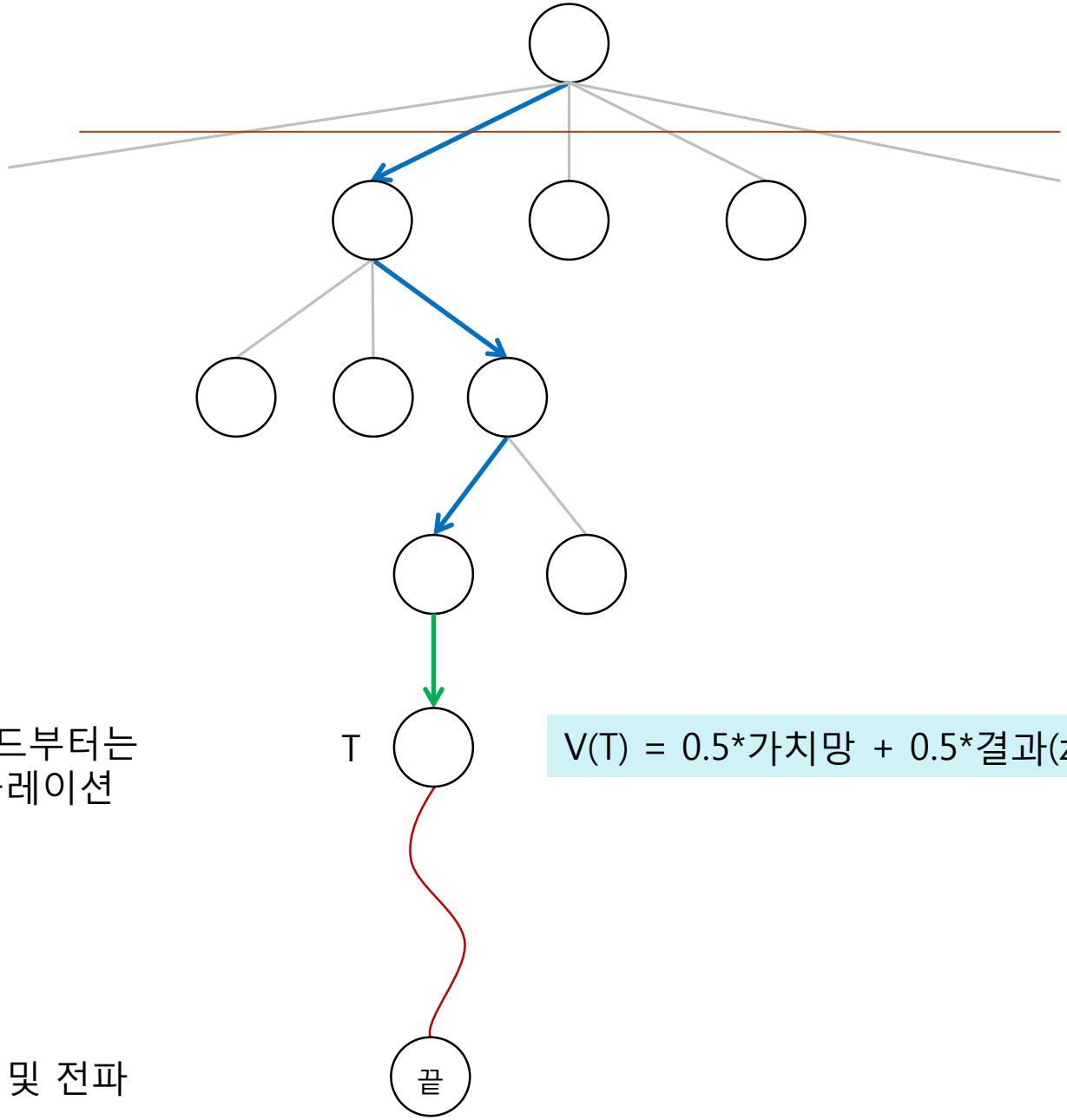
T

노드의 가치 =
가치망 (지도학습+강화학습) 계산값
+
축적된 시뮬레이션 결과



이길
가능성

노드의 가치계산 2 – 빠른 시뮬레이션 결과 반영



터미널노드부터는
빠른 시뮬레이션

V(T) = 0.5*가치망 + 0.5*결과(z)

결과확인 및 전파

액션의 가능성
변경

가치 변경

고속 실행정책

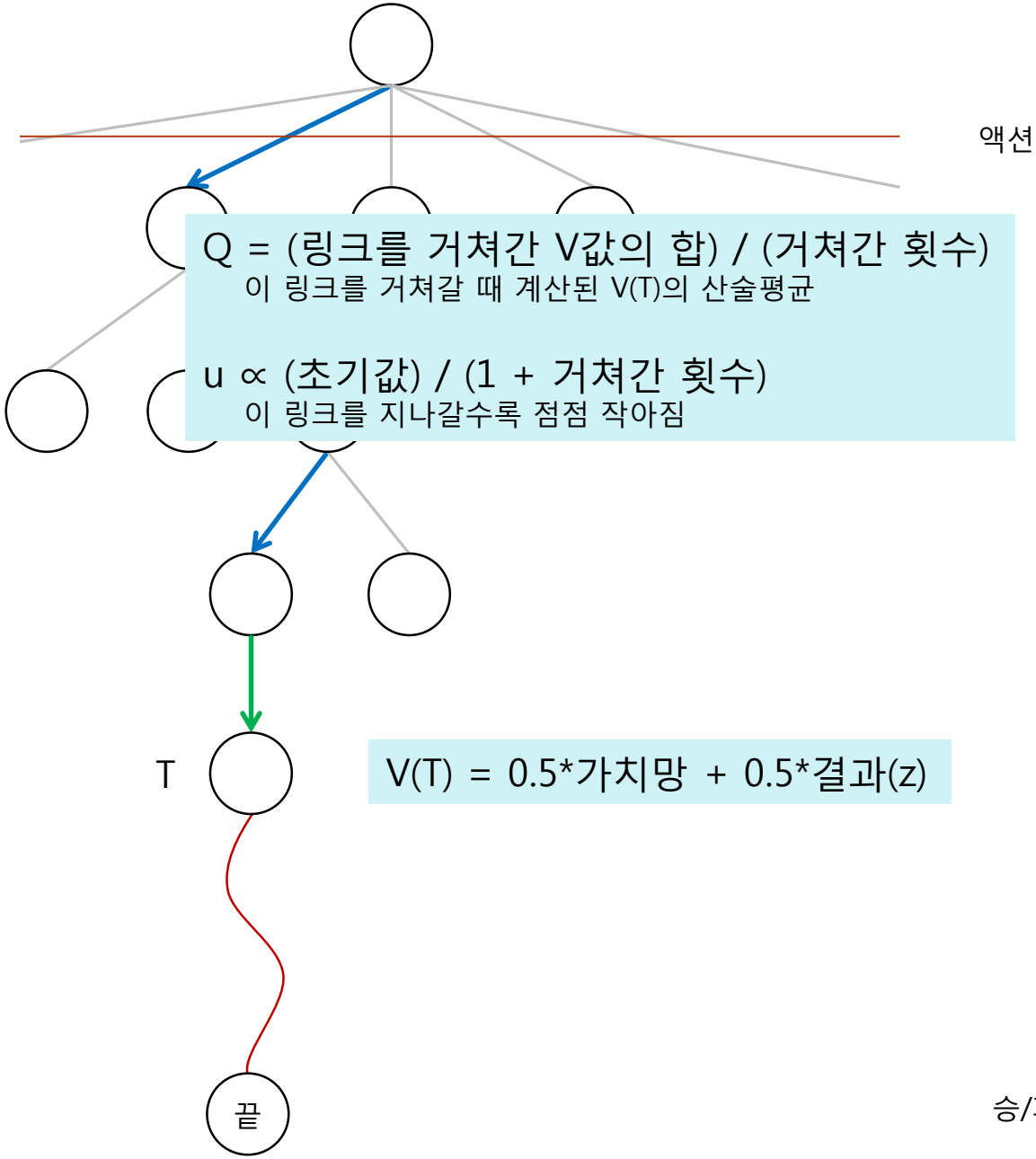
Backup

승/패 확인(z=+1/-1)

결과 Backup

Q + u
Q: 축적된 시뮬레이션 결과
u: 상태값. 시뮬레이션이 진행되며 점점 작아짐

트리탐색



터미널노드부터는 빠른 시뮬레이션

결과확인 및 전파

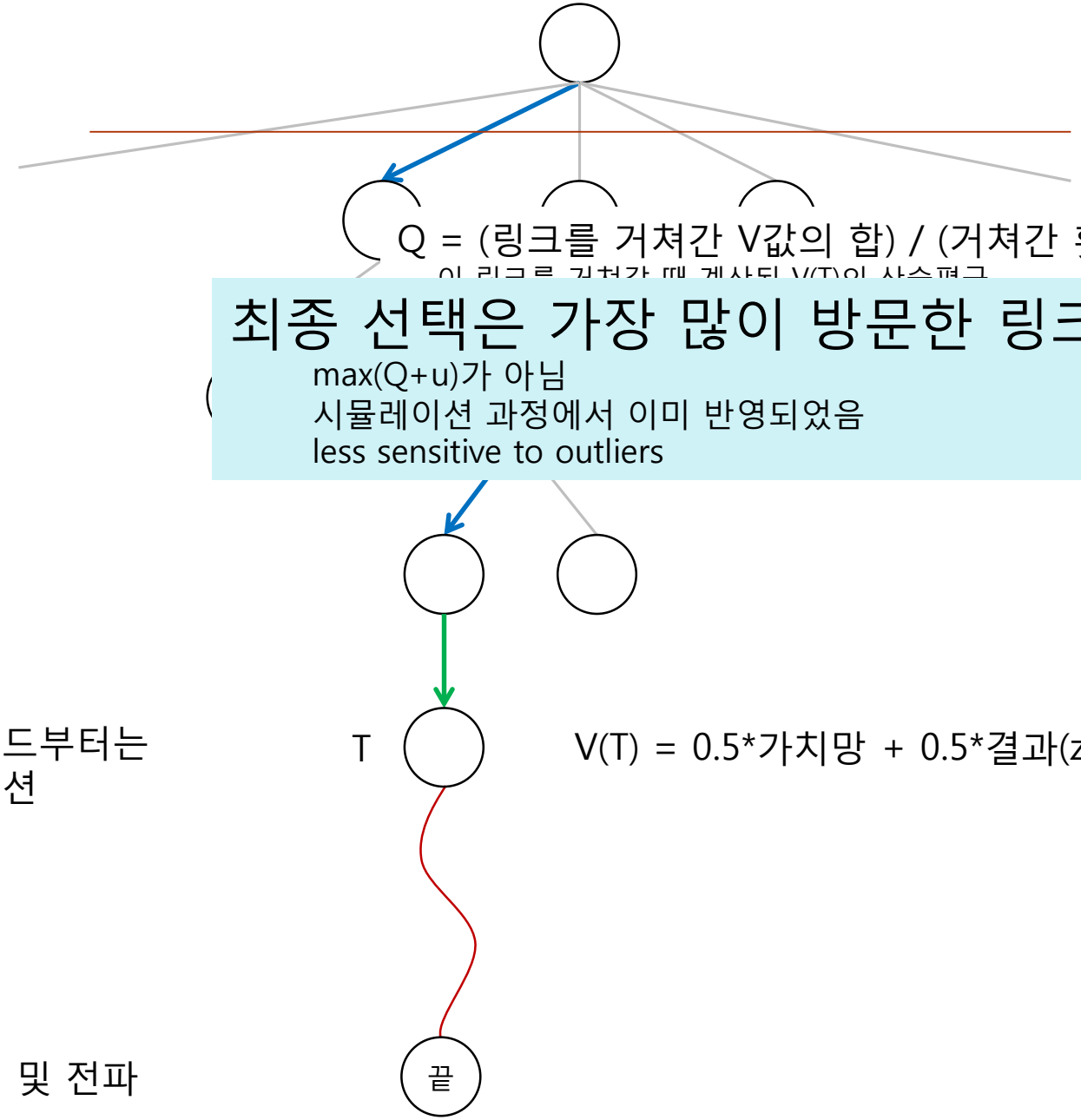
액션의 가능성 변경

가치 변경

Backup

승/패 결정

최종 선택


액션선택
Q = (링크를 거쳐간 V값의 합) / (거쳐간 횟수)
트리탐색
최종 선택은 가장 많이 방문한 링크
max(Q+u)가 아님
시뮬레이션 과정에서 이미 반영되었음
less sensitive to outliers
터미널노드부터는
시뮬레이션
T
V(T) = 0.5*가치망 + 0.5*결과(z)
결과확인 및 전파
끝
액션의 가능성
변경
가치 변경
Backup
승/패 결정

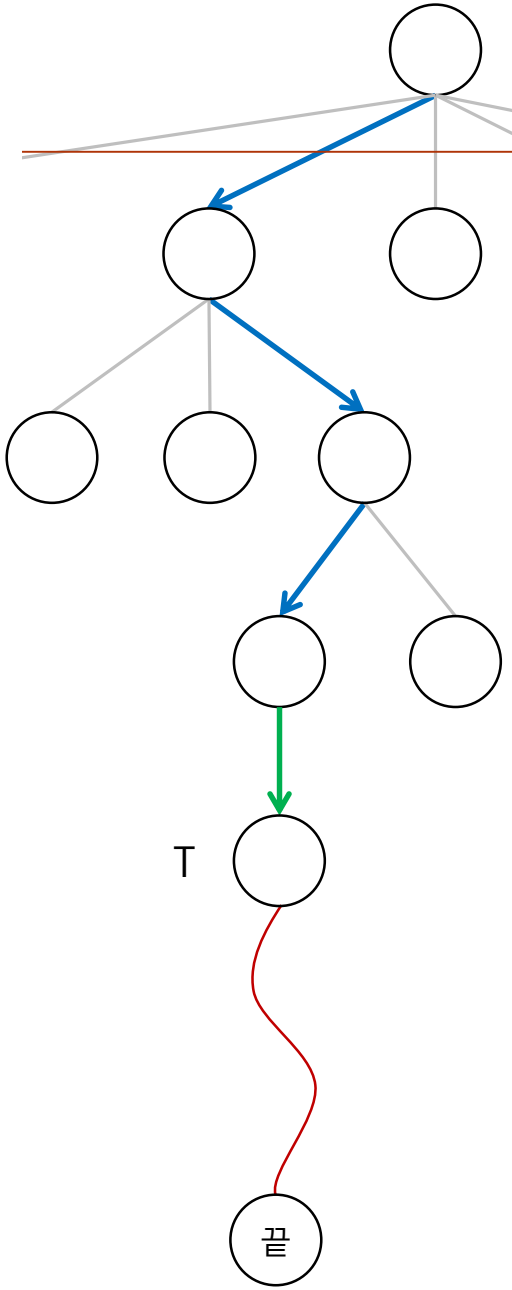
알파고의 몬테카를로 트리 탐색 요약

Q + u
Q: 축적된 시뮬레이션 결과
u: 상태값. 시뮬레이션이 진행되며 점점 작아짐

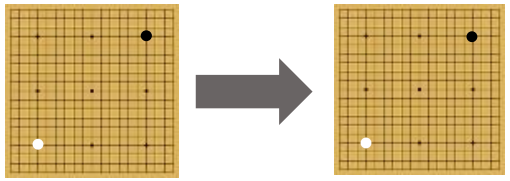
트리탐색
최대값을 따라 내려감

터미널노드부터는
빠른 시뮬레이션

결과확인 및 전파

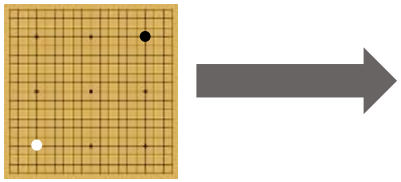


링크의 초기값을 지도학습 정책망으로 지정
많은 시뮬레이션으로 보정



다음 수의 위치별
확률분포

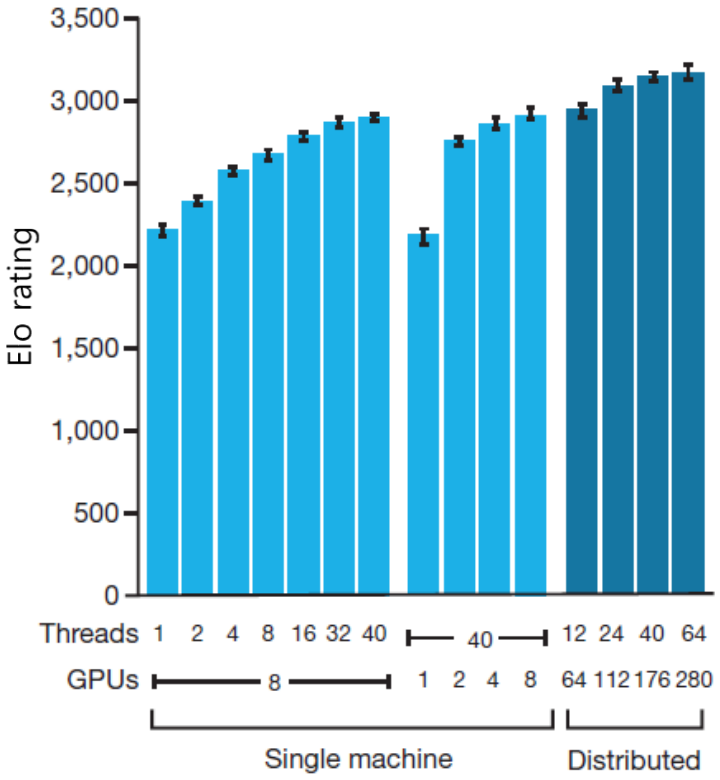
노드의 가치 =
가치망 계산값 + 시뮬레이션 결과



이길
가능성

고속 실행정책

성능 확장성 실험



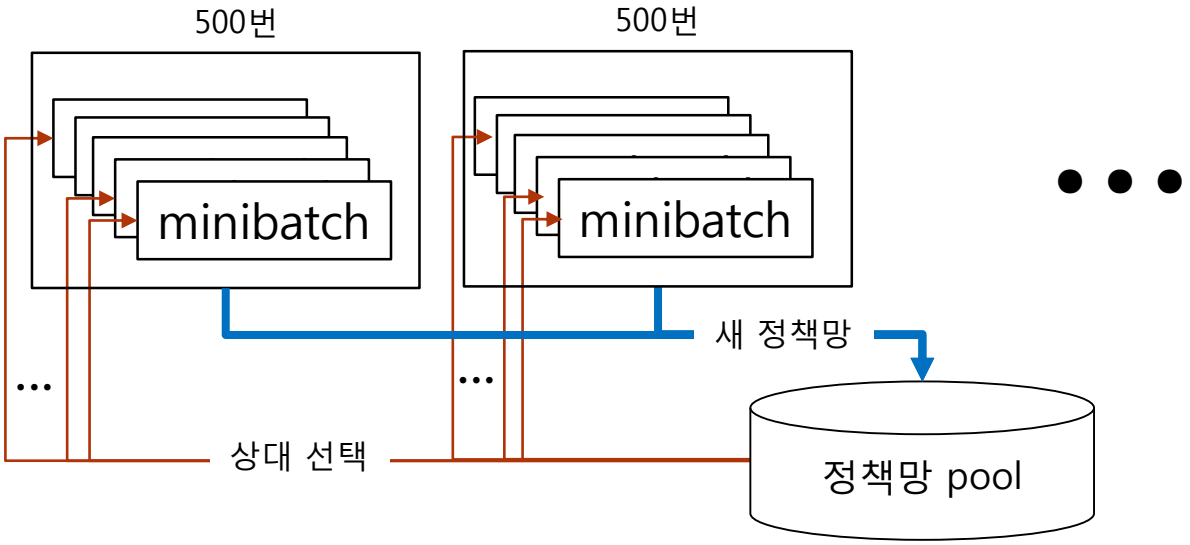
Extended Data Table 8 | Results of a tournament between AlphaGo and distributed AlphaGo, testing scalability with hardware

AlphaGo	Search threads	CPUs	GPUs	Elo
Asynchronous	1	48	8	2203
Asynchronous	2	48	8	2393
Asynchronous	4	48	8	2564
Asynchronous	8	48	8	2665
Asynchronous	16	48	8	2778
Asynchronous	32	48	8	2867
Asynchronous	40	48	8	2890
Asynchronous	40	48	1	2181
Asynchronous	40	48	2	2738
Asynchronous	40	48	4	2850
Distributed	12	428	64	2937
Distributed	24	764	112	3079
Distributed	40	1202	176	3140
Distributed	64	1920	280	3168

Each program played with a maximum of 2 s thinking time per move. Elo ratings were computed by BayesElo.

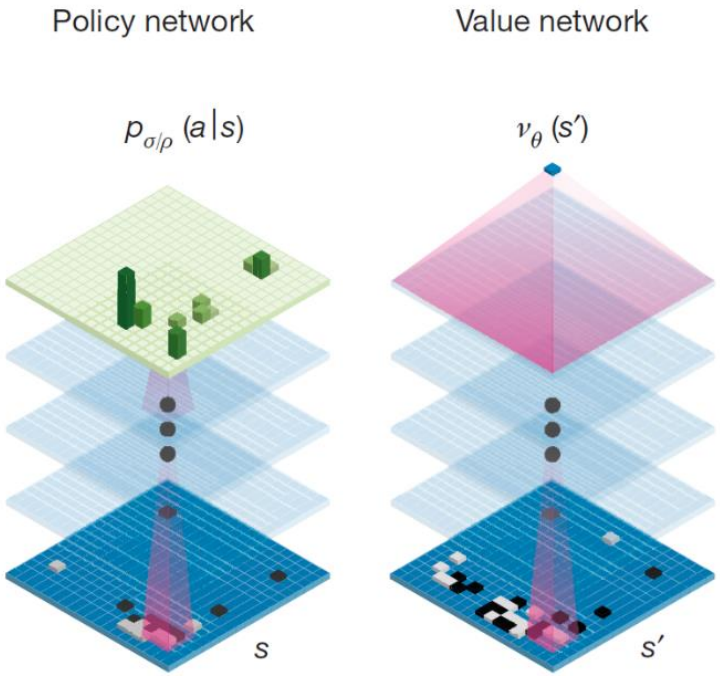
강화학습

- 지도학습의 결과를 자체 경기를 통해 강화
- 1 minibatch
  - X: 훈련대상, Y:임의의 상대(정책망)를 pool에서 선택
  - 128번 경기 후 결과의 평균을 X값에 반영-이기는 길을 강화, 지는 길을 약화
- 500번의 minibatch후 X를 새로운 정책망으로 pool에 추가
- 1주일간* 50개의 GPU로 10,000번의 minibatch를 실행하여 강화학습



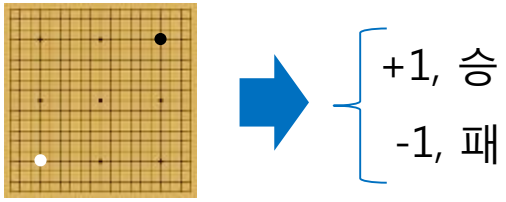
* : 논문에 1 day로 나오지만 실버교수 발표에서는 1 week 이라고 함. 아마 논문의 typo 일 듯.

가치망 훈련



학습데이터 필요

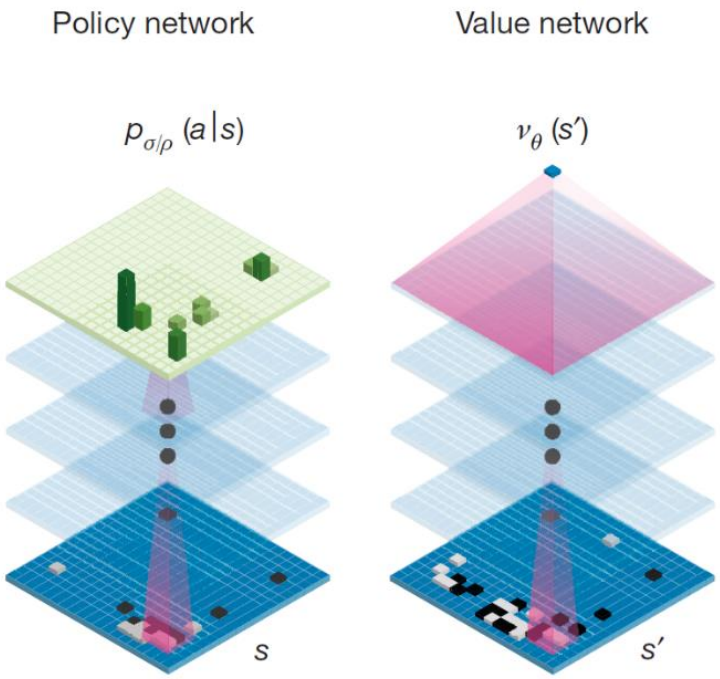
[게임상태(S'), 결과(승/패)]의 쌍



+1, 승

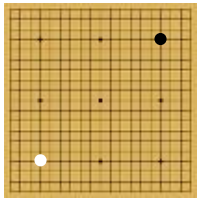
-1, 패

가치망 훈련-강화학습정책망에 기반



학습데이터 필요

[게임상태(S'), 결과(승/패)]의 쌍



+1, 승

-1, 패

1~450에서 임의 숫자(U) 선택
– 몇번째 수에 대한 데이터를 만들지 결정

1~U-1까지 지도학습정책망으로 실행

데이터셋의 다양성을 높이기 위해 noise가 더 많은 정책망 이용

U번째 수는 현재 가능한 착점 위치중 임의선택

기존 데이터셋과의 연관관계 제거

U+1~끝까지는 강화학습정책망으로 실행

게임결과는 z값(승 +1, 패 -1)으로 기록

(S_{U+1}, Z)가 하나의 학습데이터가 됨

U+1~끝까지는 강화학습정책망의 가치값 함수 추출

위의 과정을 3천만번 반복하여 3천만 데이터 확보

학습은 32개 위치의 미니배치 5천만건을 1주일간 50개의 GPU로 실행

1, 2, …, U-1, U, **U+1**, …, N(종료)

지도학습정책망으로 실행 | 랜덤 | 강화학습정책망으로 실행

알파고의 알고리즘

- **몬테카를로 트리탐색에 컨벌루션 신경망을 적용하여 성능 향상**
  - 입력데이터는 KGS 바둑서버의 3천만가지 수 (6단이상의 약 16만개의 경기)
  - 13단계의 깊은 신경망을 구성하여 막대한 양의 입력데이터 정보를 모델링
    - (cf.) 페이스북의 DeepFace가 9단계였음
- **정책망으로 액션(링크)의 초기 확률을 계산**
  - 지도학습 정책망을 사용하여 입력데이터에서 많이 선택된 수가 높은 초기값을 갖게 됨 (판후이 대국)
  - 현재는 강화학습 정책망을 사용할 수도 있음
- **가치망으로 상태(노드)의 승리확률의 절반을 계산**
  - 지도학습+강화학습의 결과를 이용하여 바둑판 상태의 승리확률을 모델링
- **시뮬레이션이 계속되며 초기값의 비중이 작아짐**
  - 링크값, 가치값이 점점 시뮬레이션 결과를 반영

경기관련 참고 사항

- 제한시간은 주로 중반전에 많이 배정
- 초읽기는 반드시 마지막까지 이용하고 착수
- 상대가 생각할 때도 시뮬레이션은 계속됨
- 30초에 약 10~20만번의 시뮬레이션을 수행할 것으로 추정

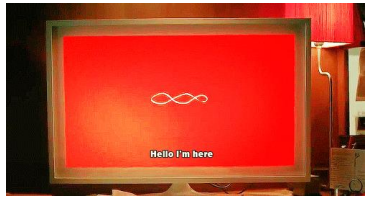
마무리 (before games)

- 제한시간과 초읽기를 2배로 한 것은 누구에게 유리할까?
  - 시뮬레이션의 비중
- 중국식 룰을 쓸 수 밖에 없는 이유
  - 개발팀에서도 얘기했듯이 7.5집 덤이라는 규칙이 정책망, 가치망에 녹아 들어 있다
  - 규칙을 바꾸면 다시 훈련시켜야 하고 지금 수준이 되려면 아마 수개월이 걸릴 것이다
- 보수적, 침착, 벽에다 두고 두는 느낌(판후이)

- 이 알고리즘은 바둑용이 아니라 복잡한 요소를 고려하여 의사결정하는 문제를 푸는 방법
  - 의료진단, 금융 등
  - 승부가 중요하지 않음. 이번에 이기더라도 인간보다 바둑을 잘 둔다는 의미도 아님
- 이세돌9단의 기보를 훈련하는 것보다 쓰레드 오류, 네트워크 오류 등 오류 상황 대응기능을 보완하여 시뮬레이션 횟수를 늘리는 것이 중요
  - GPU나 쓰레드에 오류가 있어서 전체 컴퓨팅 성능을 활용하지 못하게 되면 알파고의 수준이 훨씬 떨어짐
  - 이세돌과의 경기전 인터뷰에서 하싸비스는 알고리즘 개선에 가장 많은 노력을 들였다고 함 – 이게 무슨 의미일까요?

알파고에 대한 오해(after games)

- 인공두뇌, 트랜센던스, 공각기동대, Samantha(Her) 같은 이미지
  - 알파고도 프로그램이다
- 신경망에 바둑판을 입력하면 다음 수의 확률이 신경망 출력에 나온다
  - 최종 확률은 시뮬레이션 결과로 나온다
  - 신경망은 MCTS 시뮬레이션의 요소
- 스스로 학습한다, 알아서 밤새 강화학습하고 훈련한다
  - 사람이 명령을 내려 학습 진행
  - 신경망구성, 학습데이터 선별, 강화학습 진행에 연구자가 지속적으로 개입
- 4국 78수는 버그
  - 알파고 소프트웨어의 품질이다
- 이번 대국은 인간과 기계의 대결이다
  - 알파고의 벤치마크테스트이다




```
                $ cat test.sh
#!/bin/sh

shopt -s expand_aliases

alias lldir='ls -la'

lldir
                $ ./test.sh
total 45
drwx------   4        users  312 Oct 15 16:24 .
drwxrwxrwt  20        root   472 Sep  7 11:14 ..
-rwx------   1        users  346 Oct  3 21:26 .alias
-rwx------   1        users 5312 Oct 15 13:30 .bash_history
-rwx------   1        users  836 Oct  3 21:31 .bash_profile
-rwx------   1        users 1290 Jun 19 15:25 .bashrc
-rwx------   1        users  375 Jun 19 15:25 .cshrc
drwx------   2        root   200 Oct 11 10:25 .ssh
-rwx------   1        users 9308 Oct 15 16:22 .viminfo
drwx------   2        users  200 Oct 15 12:12 bin
-rw-r--r--   1        users  164 Oct 10 11:00 hostsfile
-rwxr-xr-x   1        users   64 Oct 15 16:22 test.sh
                $
```