위의 메모리 사용량을 보면 알다시피 A\*알고리즘은 Local search 알고리즘인 Hill climbing보다 15 \* 15 = 225배나 작은 영역에 대한 검사를 하는데도 메모리 사용량이 400배나 되는 것을 볼 수 있다.

A\*알고리즘은 탐색하는 모든 노드의 상태를 저장하는 특성을 가지고 있어서, 검사 영역이 커지면 확인해야하는 노드의 상태가 기하급수적으로(size의 제곱만큼) 증가하는데 그 많은 노드들을 전부 저장하고있어야 하고 우선순위큐로 휴리스틱의 관리도 해야하므로 메모리를 어마어마하게 잡아먹는다.

반면에 Hill climbing은 랜덤하게 값을 설정한 하나의 보드의 상태에서 파생 가능한 size \* (size -1)개의 보드의 상태 중 휴리스틱이 가장 낮은 상태 하나만 뽑아 다음 처리과정을 진행하면 되므로 메모리 사용량이 고정적이다.

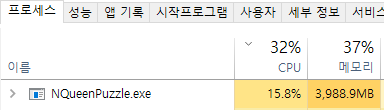
또한, 실행화면 캡쳐사진을 보면 A\*알고리즘이 size 10일 때 7.59초 동안 굉장히 빠른 속도로 24681120번의 노드 상태를 확인하여 답을 찾았지만,

Hill climbing 알고리즘은 size 50일 때 0.18초 동안 A\*에 비해서는 굉장히 느린 속도로 size \* (size -1)개의 후행자들 중 휴리스틱이 가장 작은 노드의 상태를 33번 확인하여 답을 찾았다.

위의 결과들을 봤을 때 적어도 퀸 퍼즐과 같은 ‘완전 상태 형식화’를 사용하는 상황에서는 A\* 알고리즘보다 Local search가 압도적인 성능을 보인다는 것을 알 수 있다.



**Hill climbing의 메모리 사용량 5mb~20mb (board size: 150)**



**Astar의 메모리 사용량 ~4GB (board size: 10)**