그라프에서의 휴리스틱 탐색에 관한 연구

김 명 재[†] · 정 태 충[††]

요 약

A*와 같은 Best-first 휴리스틱 탐색 알고리즘들은 인공지능 분야에서 많은 문제를 해결하는데 가장 중요한 기법들 중의 하나이다. 휴리스틱 탐색의 공통적 특성은 계산의 복잡도가 매우 높다는 것이며, 이는 수많은 노드를 가진 지도에서 경로를 찾는 것과 같은 실질적인 문제 영역에 적용되기 어렵다는 것을 나타낸다.

본 논문에서는, 몇몇 휴리스틱 탐색 알고리즘이 언급되고, path-sensitive heuristic이라 불리는 새로운 동적 가중치 휴리스틱 방법이 제안되었다. 이 방법은 동적 가중치 휴리스틱에 기초하였고, 동적 휴리스틱은 admissible heuristic을 허용하지 않거나 휴리스틱의 정확도가 떨어지는 실제 문제 영역에서 탐색 노력을 줄이는데 사용될 수 있다. 탐색 과정 동안 ω(가중치)가 동적으로 조정된다는 점에서, 다른 동적 가중치 휴리스틱 알고리즘과 구분된다.

A Study on the Heuristic Search Algorithm on Graph

Myoung-Jae Kim[†] · Tae-Choong Chung[††]

ABSTRACT

Best-first heuristic search algorithms, such as A* algorithm, are one of the most important techniques used to solve many problems in artificial intelligence. A common feature of heuristic search is its high computational complexity, which prevents the search from being applied to problems in practical domains such as route-finding in road map with significantly many nodes.

In this paper, several heuristic search algorithms are concerned. A new dynamic weighting heuristic method called the path-sensitive heuristic is proposed. It is based on a dynamic weighting heuristic, which is used to improve search effort in practical domain such as admissible heuristic is not available or heuristic accuracy is poor. It's distinctive feature compared with other dynamic weighting heuristic algorithms is path-sensitive, which means that ω(weight) is adjusted dynamically during search process in state-space search domain. For finding an optimal path, randomly scattered road-map is used as an application area.

1. 서 론

그라프에서 해를 찾는 알고리즘으로 이용될 수 있는 A*는 Hart, Nilsson과 Raphael[HN68]에 의해 정의되었으며, 목적 노드와 시작 노드에서의 양방향 탐색 알고리즘은 Pohl[Po69, Po71]에 의해 연구되었고, Pohl의 알고리즘을 개량한 BHFFA는 de Champeaux and Sint[CS71]에 의해 연구 되었다. Sparse network에서 효과적인 알고리즘은 Tarjan[Ta81]에 의해 연구되었고, 구현에 관한 기법은 Veren과 Jansen[VJ87]의 연구에서 소개되었다.

그라프에서 goal을 찾는 문제에 있어서, 알맞은 조건하에서는 A*의 성능이 다른 유사한 admissible 알고

† 준 회 원:경희대학교 전자계산공학과
†† 정 회 원:경희대학교 전자계산공학과
논문접수:1997년 1월 29일, 심사완료:1997년 9월 22일

리즘 보다 최적에 더 가깝다. 그러나 문제의 조합이 탐색된 해의 평가 값 보다 탐색 효율의 최소화에 더 연관되어 있을 때 nonadmissible 휴리스틱 알고리즘들이 사용될 수 있다. Nonadmissible 휴리스틱중 대표적인 것은 Pohl에 의해 일반화된 dynamic weighting 휴리스틱, Harris의 band-width 휴리스틱등이 있다. Pohl의 dynamic weighting 휴리스틱은 solution path의 quality를 완화하여 탐색 효율을 높일 수 있으나 적당한 weight값을 결정하기가 힘들며, Harris의 band-width 휴리스틱에서도 band-width의 두 가지 조건을 만족하는 휴리스틱을 찾기가 힘들다.

이에 본 논문에서는, 휴리스틱 탐색에서 최적화를 완화하여 탐색 효율을 높이는 dynamic weighting 기법의 한 방법이며, 탐색 과정 중 지나온 arc정보를 이용하여 쉽게 휴리스틱을 구할 수 있는 path sensitive 휴리스틱 방법을 제안, 논의해보고, 그 성능을 실험을 통하여 다른 nonadmissible 휴리스틱 알고리즘들과 비교 분석해 본다. 또한 기존의 대표적인 순서 상태 공간 탐색 방법들을 구현, 그 성능을 비교해본다.

2. 휴리스틱 탐색

2.1 휴리스틱 탐색의 개요 및 A* 알고리즘

휴리스틱이란 문제의 goal에 이르기 위한 여러 가지 방법 중 가장 효과적으로 보이는 방법을 선택하기 위한 조건, 방법, 혹은 성질(property)을 말한다. 이러한 휴리스틱 정보를 이용한 탐색 방법을 휴리스틱 탐색이라 말한다.

탐색과정중 휴리스틱 정보는

①다음에 확장될 노드의 선택

②확장을 할 때 successor의 선택

③pruning 될 노드의 결정

등에 이용되며 탐색 과정을 효과적으로 guide한다.

A* 알고리즘은 state-space 그라프에서 start노드와 goal노드를 연결하는 가장적은 비용의 경로를 찾기 위해 Hart, Nelson and Rafael[HN68]에 의해 정의되었다. A*의 전신은 Dijkstra와 Moore의 알고리즘을 포함하며, A*와 유사한 알고리즘 부류로는 branch-and-bound라는 이름으로 OR(operation-research)에서 사용된다.

또한 A*는 순서화된 상태 공간 탐색이며, 구별되는 특성은 평가 함수 f*(f*는 f의 예측값을 의미한다)의 사용이다. 일반적인 순서화된 탐색처럼 확장을 위해 선택되는 노드는 f*가 최소인 것이다. f*는 다음과 같이 정의된다.

$f^*(n) = g(n) + h^*(n)$

g(n) : start에서 노드 n까지의 최소 비용

h*(n): 노드 n에서 goal까지의 최소예측비용

그러므로 f*(n)의 값은 노드 n을 통과하는 탐색 경로의 최소 예측 값을 포함한다. 실제 비용은 f, g, h에 의해 표시되며 f*, h*는 단지 예측 값이다. 확장하기 위해 고려된 노드 n에 적용된 g 함수는 알고리즘으로 발견된 start에서 노드 n까지의 가장 작은 비용 경로의 실제 값이다. h*는 휴리스틱 함수로서 정보의 전달자이고 문제에 따라 알맞게 정의될 수 있다. A* 알고리즘에서 흥미있는것은 h*가 양수가 되어야 하며, 결코 실제 거리보다 크게 예측되지 않는다는 것이다. 즉, 임의 노드 n에서 목표까지의 최적 경로 h(n)보다 h*(n)이 작거나 같다는 것을 말한다.

2.2 최적화 요구의 완화

A*는 평가(estimate) 함수 $f^* = g + h^*$를 사용하는 상태 공간 탐색이다. h*(n)을 h(n)보다 작거나 같게 평가하면서 적절한 조건(admissibility condition)을 만나면, 최적 해를 보장하게 된다.

알맞은 조건에서는, A*의 성능이 유사한 다른 admissible 알고리즘보다 최적에 더 가깝다. 그러나 여기에 몇 가지 질문을 할 수 있다.

1) 문제가 solution cost 최소화보다 탐색 효율의 최소화에 더 연관될 수 있다.
   이 경우 $f^* = g + h^*$가 적절한 평가 함수인가?
2) Solution cost가 중요하다고 할지라도,문제의 조합이 admissible A*가 끝내지 못하는 것이 있을 수 있다.
   Solution quality를 어느 정도 완화시켜 속도면에서 이득을 얻을 수 있는가?
3) Admissible 조건을 만족시키는 h*를 찾기 어려울 수 있다.
   좋지않은 admissible 휴리스틱 함수를 사용하면 A*는 blind 탐색으로 축소된다.
   탐색이 nonadmissible 휴리스틱 함수에 의해 어떤 영향을 받는가?

첫번 질문의 답변은 다음과 같다. 평가 함수에 g를 포함시키는 이유는 탐색에 breadth-first 요소를 추가시키는 것이다. g가 없으면 평가 함수 f*는 노드 n에서 goal까지 남은 거리를 평가하고 노드 n까지 이미 발견된 거리를 무시할 것이다. 목적이 solution cost의 최소화보다 탐색 효율의 최소화에 있다면 g가 생략될 수 있다고 말할 수 있다. 이같이 수행되는 초기의 휴리스틱 탐색 알고리즘은 그라프 운행 알고리즘이었다. 평가 함수의 형태는 $f^*=h^*$이고 8 puzzle과 다른 문제들의 해결에 있어 탐색 효율의 최소화에 사용했다. 이 GTA(Graph Traversal Algorithm)를 일반화한 것과 A*, 그 밖의 것들은 HPA(Heuristic Path Algorithm)로 정의되며 아래의 평가 함수로 순서화된 state-space 탐색을 행한다.

$f^*=(1-w)g+wh^*$

w:[0, 1]의 weighting

w=1.0:GTA, w=0:bredth-first 탐색,

w=0.5:A*로서 $f^*=g+h^*$가 된다.

Pohl의 연구에 의하면 이 HPA들은 특별한 경우에도 g를 생략하는 것이 잘못임을 보인다. 한 경우는 h^*가 가장 정확한 휴리스틱 함수일 경우이다; 모든 노드에서 $h^*=h$이면, 1)$f^*=h^*$는 2)$f^*=g+h^*$ 보다 적지않은 노드를 확장한다. 다른 경우는, 무한 m-ray 트리인 간결화된 state-space를 가정하고, h^*의 오류가 양의 정수 ε에 의해 bound된다고 가정하자. 이때 식 1)로 확장된 최대 노드 수는 식 2)보다 크며 그 차이는 오류 한계 ε내에서 지수적이다. 두 함수의 average-case behavior는 분석되지 않았으나, 경험적 결과는 ordered 탐색이 가장적은 수의 노드를 확장하고 g를 포함하되 h^*보다 가중치($0.5<w<1$)가 적게 주어지면 h^*가 좋다고 할 수 있음을 보여준다.

두번째는 속도와 solution quality에 관계된 질문으로, Pohl은 HPA의 휴리스틱 함수를 일반화했다; $f^*(n)=g(n)+w(n)h^*(n)$로 weight가 상수가 아니다. w(n)은 1보다 크거나 같으며, 노드 n의 깊이에 따라 변화한다. 이것을 dynamic weighting이라 부른다. 탐색이 깊이 들어갈수록 h^*의 가중치를 적게 한다. h^*가 h보다 lower bound된다는 가정을 사용해, Pohl은 HPA의 cost가 다음 비율로 bound되는 TSP(Traveling Salesman Problem)의 solution을 구할 수 있음을 보였다.

$$\frac{\text{cost of tour found}}{\text{cost of optimal solution}} < 1+\varepsilon$$

ε: weight, $\quad 0<=\varepsilon<1$

Harris의 band-width 탐색은 위와 약간 다르다. 여기서는 admissibility 조건을 만족하나 좋지않은 h^*를 유용하다 가정하는데, nongoal 노드 n에 대한 다음의 band-width 조건을 소개했다.

1) $h^*(n)\le h(n)+\varepsilon$

2) $h(n)-d\le h^*(n)$

여기서 h^*가 monotonic 가정을 만족한다고 가정한다.

A*가 goal까지의 거리를 ε보다 크게 평가하지않으면, A*가 찾은 solution cost는 ε범위 내에서 optimal solution을 넘지 않는다. 이 h^*를 사용하는 알고리즘을 ε-admissible하다 하고, 그 solution을 ε-optimal이라 한다. Band-width가 한번 goal을 찾고, 조건 (1)을 그 이상 적용하면 그 solution은 ε에 대해 optimal solution에 더 가깝다. 이것은 (a)발견된 solution을 알고있고, (b)OPEN list의 남아있는 모든 노드 n상에서 다른 모든 solution cost의 lower bound가 $f^*(n)-\varepsilon$의 최소값이기 때문에 가능하다. 두 양의 차이가 크면, 탐색은 optimal에 가까운 solution을 찾을 때까지 계속 될 수 있다. band-width의 조건(2)는 OPEN list에서 실제 optimal path상의 노드가 아닌 노드를 *dropping*하여 기억 공간을 절약하는데 사용될 수 있다. q를 확장을 위해 선택된 f^*의 최소값을 갖는 노드라하고, $f^*(m)$이 $f^*(q)$보다 크면 노드 m을 OPEN list에서 dropping한다. 특히, 모든 노드 m은 노드 q가 다음과 같으면 dropping된다.

$f^*(m)-(\varepsilon+d)>f^*(q)$

Harris는 band-width의 두 조건을 만족하는 h^*를 발견하기 어렵다는 것을 알았다. 그러므로, h^*를 대신하여 탐색을 순서화하고 어느 노드가 dropping 될 것인가를 결정하는 두 휴리스틱 함수를 정의할 수 있을 것이다.

이 함수들을 다음을 만족하는 h_1^*, h_2^*라 할 수 있다.

(1') $h_1^*(n)\le h(n)+\varepsilon$

(2') $h(n)-d\le h_2^*(n)$

3. Path-sensitive 휴리스틱

3.1 Path-sensitive 휴리스틱

일반적으로 휴리스틱 함수의 정확도인 $h^*(n)/h(n)$이 1에 가까워질수록 확장되는 노드 수는 감소하며 탐색의 효율은 좋아진다. 어떤 문제의 admissible한 휴리스틱을 구하는 것은 어려운 문제이며, 비록 admissible해도 정확도가 너무 낮으면 탐색 효율은 blind 탐색에 비해 별로 나아질 것이 없다.

본 절에서는 overestimate될 가능성이 있지만 탐색을 하는 중에 admissibility를 높임으로써 탐색의 질을 높이는 dynamic weighting 휴리스틱의 한 방법을 소개한다.

먼저 설명에 필요한 정리와 그에 필요한 정의를 하자.

[정의 1] 휴리스틱 함수 h(n)이 다음 조건을 만족하면 monotone 하다고 한다.

$h^*(n) \le c(n, m) + h^*(m), \forall n, m | m \in SUCCESSOR(n)$

[정리 1] 그라프 G에서 h_1^*이 monotone하고 admissible하다면

$R = \min(c(n, m)/h_1^*(n, m)), \forall n, m \in G$

$h_2^* = R \cdot h_1^*(x, y), \forall x, y \in G$

h_2^* 또한 admissible한 휴리스틱이다.

여기서 $c(n, m)$ 은 n, m사이의 비용.

[증명] $h_2^*(x, y) \le k(x, y), \forall x, y \in G$임을 보이자. 여기서 $k(x, y)$는 x, y 사이의 실제 비용이다.

x, y 사이에 m개의 노드가 있다고 하고 그 노드들의 집합을 M이라 하면 $M \subset G$ 이고

$k(x, y) = c(n_1, n_2) + c(n_2, n_3) + \cdots + c(n_{m-1}, n_m)$,

$n_1, \ldots, n_m \in M$

R의 정의와 $h_1^*(n, m) \ge 0, \forall n, m \in G$에 의해

$$c(n, m)/h_1^*(n, m) \ge R$$
$$c(n, m) \ge h_1^*(n, m) R, \forall n, m \in M \quad (1)$$

h_1^*이 monotone하므로

$$h_1^*(x, y) \le c(n_1, n_2) + h_1^*(n_2, y)$$
$$\le c(n_1, n_2) + c(n_2, n_3) + h_1^*(n_3, y)$$
$$\vdots$$
$$\le h_1^*(n_1, n_2) + \cdots + h_1^*(n_{m-1}, n_m) \quad (2)$$

그러면

$$h_2^*(x, y) = h_1^*(n, m) \cdot R$$
$$\le R(h_1^*(n_1, n_2) + \cdots + h_1^*(n_{m-1}, n_m)) \text{ by (1)}$$
$$\le c(n_1, n_2) + \cdots + c(n_{m-1}, n_m) \text{ by (2)}$$
$$= k(x, y)$$

그러므로 h_2^*는 그라프 G상에서 admissible하다.

[증명끝]

일반적으로 노드 s에서 t까지의 경로를 그라프 G상에서 찾는 경우 G상의 모든 노드가 확장되는 것은 아니다. G상에서 admissible 휴리스틱 h^*에 의해 확장되는 노드의 집합을 M이라 하고 다음의 두 휴리스틱 h_1^*, h_2^*를 생각해보자.

$h_1^*(x, y) = h^*(x, y) \cdot R_1$, $R_1 = \min(c(n, m)/h_1^*(n, m))$, $n, m \in G$

$h_2^*(x, y) = h^*(x, y) \cdot R_2$, $R_2 = \min(c(n, m)/h_1^*(n, m))$, $n, m \in M$

이라 하면 [정리 1]에 의해 h_1^*은 G상에서 admissible하고 h_2^*은 M상에서 admissible하다.

만약 $R_1 < R_2$이면 $h_1^* < h_2^*$이고 그래프 M상에서는 h_1^*을 사용하는 것 보다 h_2^*를 사용하는 것이 더 효과적이다. 그러나 확장될 노드들의 집합 M을 미리 알 수 없기 때문에 h_2^*를 구하는 것이 어려운 일이다. h_2에 대한 approximation 함수인 h_3^*을

$h_3^*(x, y) = h^*(x, y) \cdot R_3$

$R_3 = \min(c(n, m)/h^*(n, m)), \forall n, m \in OPEN \cup CLOSED \subset M$

라 하면 h_3^*는 탐색수행과정중 dynamic하게 수정되며 결국 h_2^*에 근사하게 된다. 이러한 h_3는 수행과정중 쉽게 구할 수 있으며 최악의 경우에

$\varepsilon_{max} = h^*(s, t) \cdot (\max(c(i, j)/h^*(i, j) - \min(c(n, m)/h^*(n, m)),$
$\forall i, j, n, m \in M$

$h_3^* \le h^* + \varepsilon \ (0 \le \varepsilon \le \varepsilon_{max})$

을 만족하는 ε-admissible 휴리스틱이다. 이렇게 수행중 h^*에 대한 weight R의 값이 변하는 휴리스틱을 path sensitive 휴리스틱 이라고 정의하자.

Planar 그라프에서의 path sensitive 휴리스틱을 이용한 탐색의 효과에 대해서 4장에서 고찰하기로 한다.

4. 성능 비교를 위한 시뮬레이션

4.1 개 요

본 장에서는 path sensitive 휴리스틱의 성능을 로드맵 환경에서 비교해본다. 탐색에 사용한 휴리스틱은 두 노드 사이의 직선거리인 Euclidian-distance이며 시뮬레이션 상황은 다음과 같다.

①그라프의 생성

균등 밀도(uniform density)를 갖고 random하게 tightly interconnected 된 planar그라프를 생성하였다. 노드의 갯수는 625개로 하였다.

②비용의 부가(cost assignment)

일반적인 휴리스틱 탐색 알고리즘들의 성능은 휴리스틱의 부정확정도(HI)에 의해 많은 영향을 받는다. 생성된 planar그라프에서 이러한 상황을 반영하기 위해 이웃하는 두 노드 사이의 비용은

$$cost(n, m) = h^*(n, m) \cdot R$$

로 정하였고, 여기서 $h^*(n, m)$은 n, m사이의 Euclidean-distance이며 R은 $1 \le R \le HI$를 만족하도록 random하게 부과하였다. HI(heuristic inaccuracy)는 두 노드 사이의 휴리스틱 정확도의 평가 기준으로 입력 파라미터로써 사용되며, 다음 조건을 만족한다.

$$h^*(n, m) \le cost(n, m) \le h^*(n, m) \cdot HI$$
$$cost(n, m)/h^*(n, m) \le HI$$

즉, 실제 비용과 휴리스틱의 비는 HI에 의해 bound된다.

③비교 방법 및 평가 요소

임의로 생성된 그라프에 각 노드 사이의 비용을 부과하고, 100회에 걸쳐 random하게 start와 goal을 선정하여 출력 파라미터의 평균을 구했다. 입력 파라미터(HI)를 변경하면서 위의 실험을 반복하였다.

〈출력 파라미터〉

㉠확장된 노드의 갯수

㉡수행에 걸린 시간

㉢찾은 경로의 비용에 대한 상대 오차 E(%): Nonadmissible 휴리스틱 탐색의 경우

$$E = \frac{\text{찾은 경로의 비용} - \text{최적 경로의 비용}}{\text{최적 경로의 비용}} \cdot 100(\%)$$

㉣최적 경로를 찾은 비율 (Nonadmissible 휴리스틱 탐색의 경우)

$$\frac{\text{최적경로를 찾는 횟수}}{\text{총 수행 횟수}} \cdot 100(\%)$$

4.2 Nonadmissible 휴리스틱 알고리즘들의 성능 비교

(1) 비교 알고리즘

지금까지 사용한 Euclidian distance 휴리스틱을 h^*라 하고 다음 두 휴리스틱 h^*_{avg}와 h^*_{ps}를 아래과 같이 정의하였다.

①$h^*_{avg} = h^* \cdot (1 + (HI - 1)/2)$

즉, h^*에 생성된 그라프의 평균적 휴리스틱 정확도를 곱하여 amplitude된 휴리스틱

②$h^*_{ps} = h^* \cdot R$

$R = \min(c(n, m)/h^*(n, m))$, n, m OPEN CLOSED

즉, 앞에서 언급한 path sensitive 휴리스틱 방법을 이용한 휴리스틱

③$h^*_{dw}(n) = h^*(n) + \varepsilon \cdot (1 - h^*(s, n)/h^*(s, d))$

$\varepsilon = HI/2$, s : start node, d : goal node

즉, dynamic weighting 휴리스틱 기법을 이용한 휴리스틱 단 방향 탐색의 경우

①A* : h^*를 사용한 A* 알고리즘

②A*-avg : h^*_{avg}를 사용한 A* 알고리즘

③A*-ps : h^*_{ps}를 사용한 A* 알고리즘

④A*-dw : h^*_{dw}를 사용한 A* 알고리즘

에 대해 비교해 본다.

양 방향 탐색의 경우

①BA* : h^*를 사용한 A* 알고리즘

②BA*-avg : h^*_{avg}를 사용한 A* 알고리즘

③BA*-ps : h^*_{ps}를 사용한 A* 알고리즘

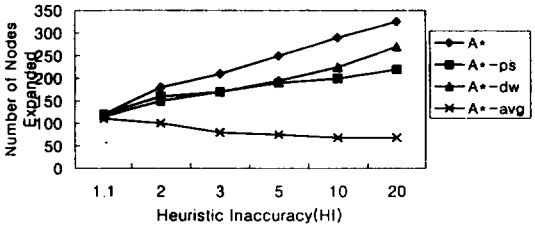
④BA*-dw : h^*_{dw}를 사용한 A* 알고리즘

⑤BFFA : de Champeaux Sint[CS71]가 제안한 알고리즘

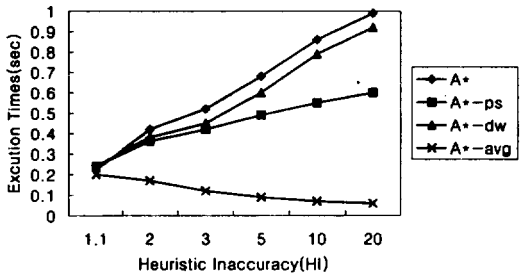
에 대해 비교해 본다.

(2)단방향 탐색에서의 성능비교

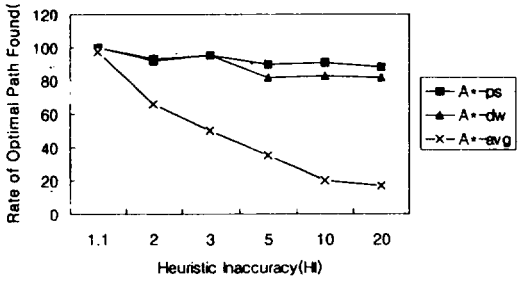
①알고리즘들의 평균 노드 확장 수



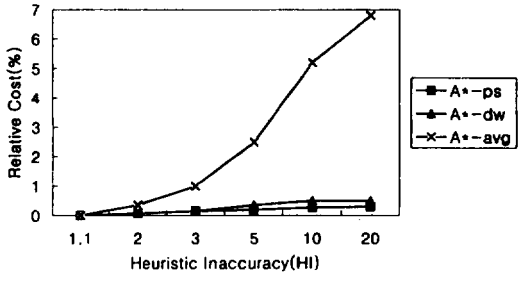
②알고리즘들의 평균수행시간



③알고리즘들의 최적 경로를 찾은 횟수

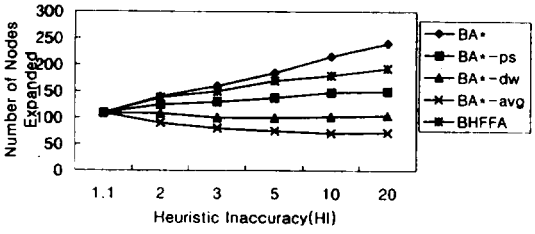


④알고리즘들의 찾은 경로의 비용에 대한 상대 오차

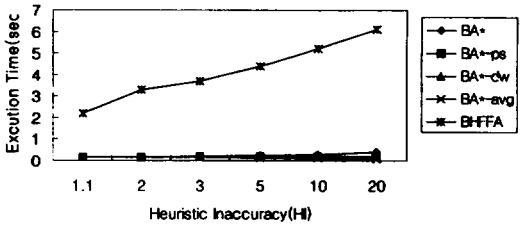


(3)양방향 탐색에서의 성능비교

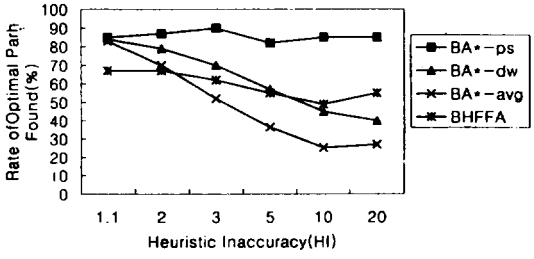
①알고리즘들의 평균 노드 확장 수



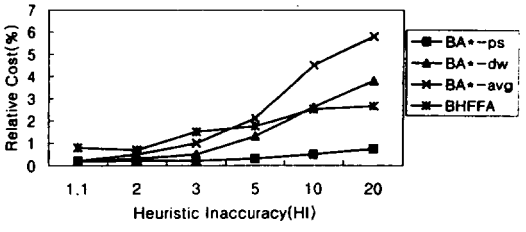
②알고리즘들의 평균수행시간



③알고리즘들의 최적 경로를 찾은 횟수



④알고리즘들의 찾은 경로의 비용에 대한 상대 오차



4.3 결과 분석

Unidirectional 탐색의 경우 amplitude된 휴리스틱을 사용한 A*-avg가 노드 확장이나 시간면에서 우수하였으나, 찾은 경로의 오차는 가장 컸다.(최고 70%)

본 논문에서 제안한 path sensitive 휴리스틱을 이용한 A*-ps는 노드 확장 수를 최고 30% 이상 줄이면서도, 90% 이상 최적 경로를 찾았으며, 이때의 비용에 대한 상대 오차는 0.4%를 넘지 않았다. 또한 A*-ps는 dynamic weighting 기법을 이용한 A*-dw보다 노드 확장면과 오차를 고려해 볼 때 우수한 것으로 나타났다. 양방향 탐색에 있어서도 다소 차이는 있어나 같은 결과를 얻었다. BA*-ps는 모든 출력 파라미터에 대해 BHFFA보다 우수한것으로 나타났다.

5. 결 론

본 논문에서는 이미 연구되었던 대표적인 휴리스틱 탐색 알고리즘들을 구현하여 그 성능을 알아보았고, 탐색과정 중 지나온 arc들의 정보에 적용하는 path sensitive 휴리스틱을 제안하여 로드맵과 같은 planar 그라프 상에서 구현해 봄으로써 탐색 효율을 평가하였다.

Planar 그라프에 적용해 본 결과 A*-ps는 노드 확장면이나 수행 시간, 찾은 경로의 최적 정도면에서 BHFFA보다 우수한 것으로 나타났다. A*-avg는 탐색 효율은 좋으나 찾은 경로의 정확도가 A*-ps에 비해 매우 떨어져으며, 실제 적용에 있어서는 amplitude될 값을 미리 알기가 어려운 단점이 있다. 또한 A*-ps는 A*-dw에 비해 노드 확장면과 오류를 고려해 볼 때 우수한 것으로 나타났다.

결과적으로 노드 수가 매우 많은 그라프에서는 구현 방법과 휴리스틱의 정확도가 매우 중요하며, 꼭 최적 경로가 필요하지 않다면 path sensitive 휴리스틱 기법을 이용하여 탐색 효율을 높일 수 있음을 알 수 있었다. 또한 path sensitive 휴리스틱 기법은 비단 planar 그라프 뿐만 아니라 임의의 state-space 탐색에 쉽게 적용되어, 휴리스틱이 admissible하지 않거나 adimissible하더라도 정확도가 매우 낮은 경우 탐색과정중 탐색 되는 state-space에 대해 능동적으로 대처할 수 있으므로 매우 효과적일 것으로 기대된다. 앞으로 이에 대한 연구가 이루어져야 할 것이다.

참 고 문 헌

[CS77] D. de Champeaux and L. Sint, An Improved Bidirectional Search Algorithm, JACM, Vol. 24. 2, April 1977, pp. 177-191.

[EM82] R. J. Elliot and Lesk, Route Finding in Street Maps by computer and people, AAAI, September 1982, pp. 258-261

[Ga81] John Gasehninig, A problem similarity approach to devising heuristics: First Results, A reading in artificial intelligence, Tioga Publishing Company, 1981, pp. 23-29

[Hs78] E. Horowitz and S. Sahni, Fundamentls of Computer Algorithm,Computer Science press, Inc. 1978.

[HN68] Hart, P.E., Nilsson, N.J., and Raphael, B. 1968. A formal basis for the heuristic determination of minimum cost paths. IEEE Transaction on SSC SSC-4: 100-107.

[Jo77] D.B. Johnson, Efficient Algorithms for Shortest Paths in Sparce Network, JACM, Vol. 24, No. 1, Jan. 1977, pp. 1-13.

[Ni71] Nilsson, N.J., Problem-Solving Methods in Artificial Intelligence, McGraw-Hill, New York, 1971.

[Pe84] Pearl, J., Heuristics, Addison Wesley, Reading, MA, 1984.

[Po69] Pohl, I., Bi-directional heuristic search in path problem. Stanford U., Calif., 1969.

[Po71] Pohl, I., Bi-directional search. Machine Intelligence 6, 1971, pp. 127-140.

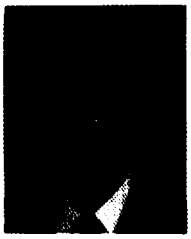
[SL89] Shierei Hvang and Larry S. Davis, Parallel Iterative A* search,IJCAI, 1989, pp. 23-29.

[Ta81] R.E. Tarjan, Fast Algorithms for Solving Path Problems, JACM, Vol. 28, No. 3, July. 1981, pp 594-614.

[VJ88] T.V. Veren and G.R.M. Jansen, Recent Developments in Path Finding Algorithms, Transportation Planning and Technology, 1988, Vol. 12, pp. 57-71.

[VL90] Vikram A. Saletore and L.V. Kale, Consistent linear sppedups to a first solution in parallel state-space search, AAAI, 1990 August, pp. 227-233.

[Wi88] P.H Winston, Artificial Intelligence, Addison-WesleyPublishing Company, 1984.



김 명 재

1989년 경희대학교 문리대 수학과 졸업(학사)
1992년 경희대학교 전자계산공학과 졸업(석사)
1992년~현재 경희대학교 전자계산공학과 박사과정 재학중
관심분야: 인공지능, 음성합성

정 태 충

1980년 서울대학교 전자공학과 졸업(학사)
1982년 한국과학기술원 전자계산학과(공학석사)
1987년 한국과학기술원 전자계산학과(공학박사)
1987년 9월~1988년 3월 KIST시스템 공학센터 선임연구원
1988년 3월~현재 경희대학교 전자계산공학과 교수
관심분야: 인공 지능, 자연어 처리, 멀티미디어 등