Sztuczna inteligencja. Przeszukiwanie z wiedzą o problemie

Paweł Rychlikowski

Instytut Informatyki UWr

10 marca 2021

Algorytm A*

Definicje

- $g(n)$ – koszt dotarcia do węzła n
- $h(n)$ – szacowany koszt dotarcia od n do (najbliższego) punktu docelowego ($h(s) \geq 0$)
- f(n) = g(n) + h(n)

Algorytm

Przeprowadź przeszukanie, wykorzystując $f(n)$ jako priorytet węzła (czyli rozwijamy węzły od tego, który ma najmniejszy f).

Algorytm A*. Uwagi

- Zwróćmy uwagę, że algorytm przypomina BFS (w którym, jak pamiętamy, używamy kolejki FIFO) oraz algorytm UCS (uniform cost search, Dijkstry).
- Jedyną różnicą między A* i UCS jest użycie funkcji f, a nie funkcji g jako priorytetu w kolejce priotytetowej.

Wymagania dla heurystyki

Oczywiście od wyboru funkcji **h** (nazywanej heurystyką) zależą właściwości algorytmu A^*

Wymienimy najważniejsze właściwości funkcji h.

1. **Nieujemna**: $h(s) \geq 0$, dla każdego s
2. **Rozsądna**: $h(s_{\text{end}}) = 0$
3. **Dopuszczalna** (admissible):
   $h(s) <$ prawdziwy koszt dotarcia ze stanu s do stanu końcowego
   Inaczej: **optymistyczna**
4. **Spójna** (consistent), s_1, s_2 to sąsiednie stany:

$$h(s_2) + \text{cost}(s_1, s_2) \geq h(s_1)$$

Ostatnia własność przypomina własność trójkąta w definicji metryki.

Popatrzmy na rysunek

O optymizmie

Pojęcie **optymistyczna** wydaje się dość intuicyjne w kontekście heurystyki.
Zwróćmy uwagę, że:

- Dla zadania: dojechać z punktu A do B po drogach publicznych, heurystyka szacująca koszt dotarcia do B jako odległość euklidesową z punktu X, w którym jesteśmy, do celu jest optymistyczna:
  zakładamy bowiem optymistycznie, że istnieje prosta, pozbawiona zakrętów droga prościutko do B
- Jeżeli podróżujemy po Manhattanie (czyli w miejscu, gdzie wszystkie drogi przecinają się pod kontem prostym), poprzednia heurystyka nadal będzie optymistyczna. Ale bardziej realistyczne będzie liczenie tzw. odległości taksówkowej, która jest po prostu sumą różnic na współrzędnych x oraz y.

(zobaczmy rysunek)

O optymizmie

Jak za chwilę zobaczymy, wybór bardziej realistycznej (ale ciągle optymistycznej) heurystyki zaowocuje lepszym działaniem algorytmu A*.

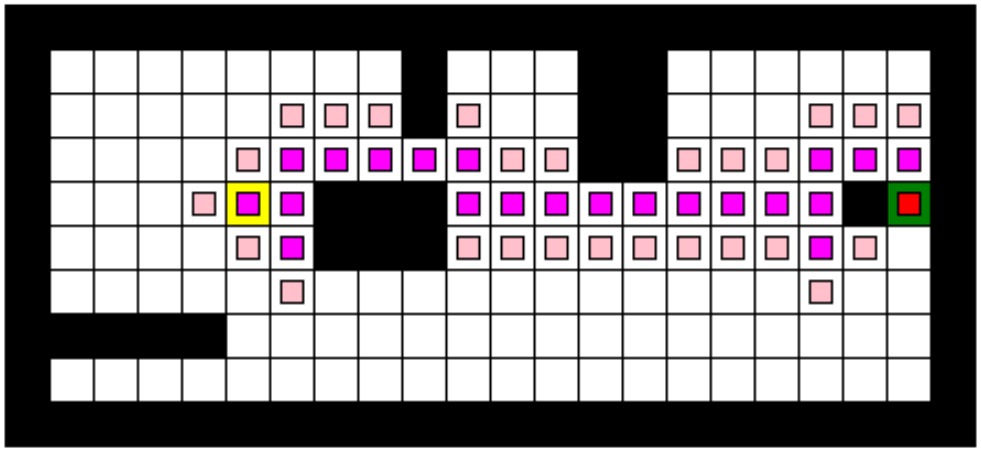
Konwencja

- Ponieważ h ma być szacowaną odległością od celu, umówimy się, że własność **nieujemności** i **rozsądności** funkcji h są konieczne, żeby mówić o algorytmie $A*$
- Pozostałe dwie własności z poprzedniego slajdu (optymistyczna i spójna) są „mile widziane”, ale niekonieczne.

Kilka prostych faktów

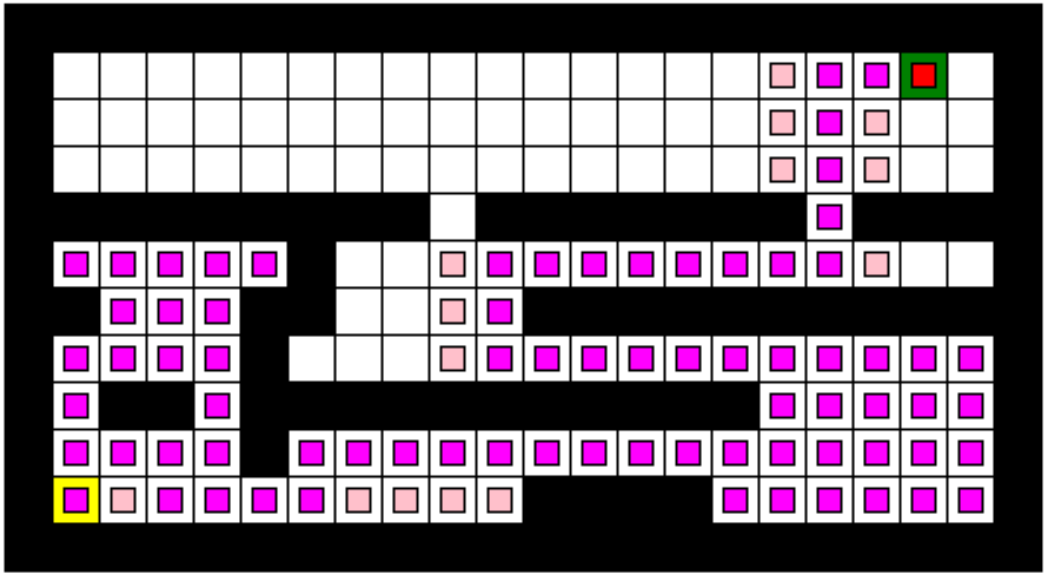
1. UCS to A^* z super-optymistyczną heurystyką (h(s) = 0)
2. Spójna heurystyka jest optymistyczna
   Dowód: Indukcja po węzłach (na ćwiczeniach, okolice C2)

A^* w labiryncie (1)



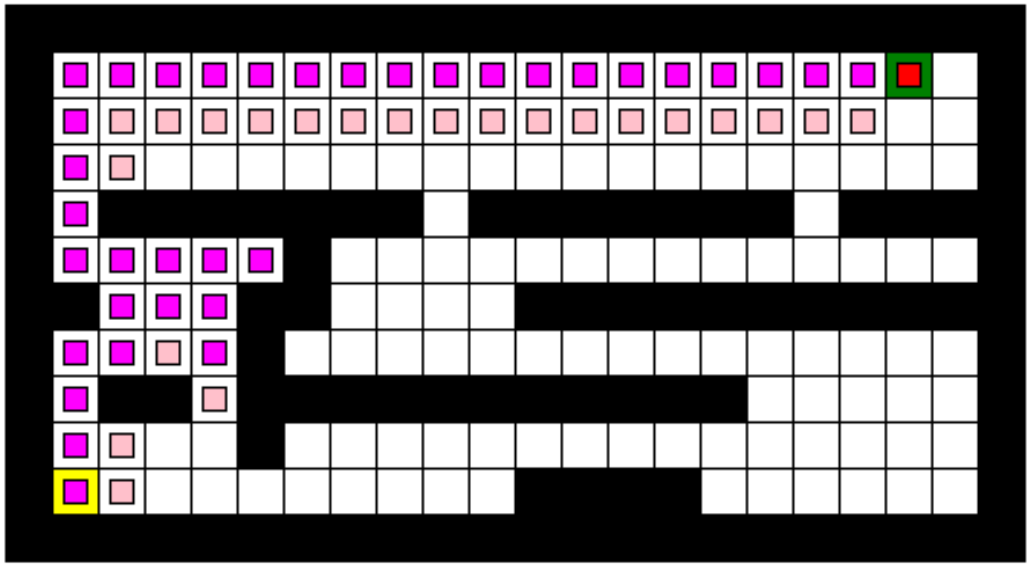
Używamy odległości taksówkowej między bieżącą kratką a celem jako heurystyki (czyli **optymistycznie** zakładamy, że nie spotkamy po drodze żadnej ściany).
Kolor różowy: węzy w kolejce, kolor purpurowy – węzły rozwinięte. Jedynie dwa rozwinięte węzły są poza optymalną ścieżka. Na tym rysunku (i kolejnych) widzimy stan algorytmu w momencie osiągnięcia węzła końcowego.

A^* w labiryncie (2)



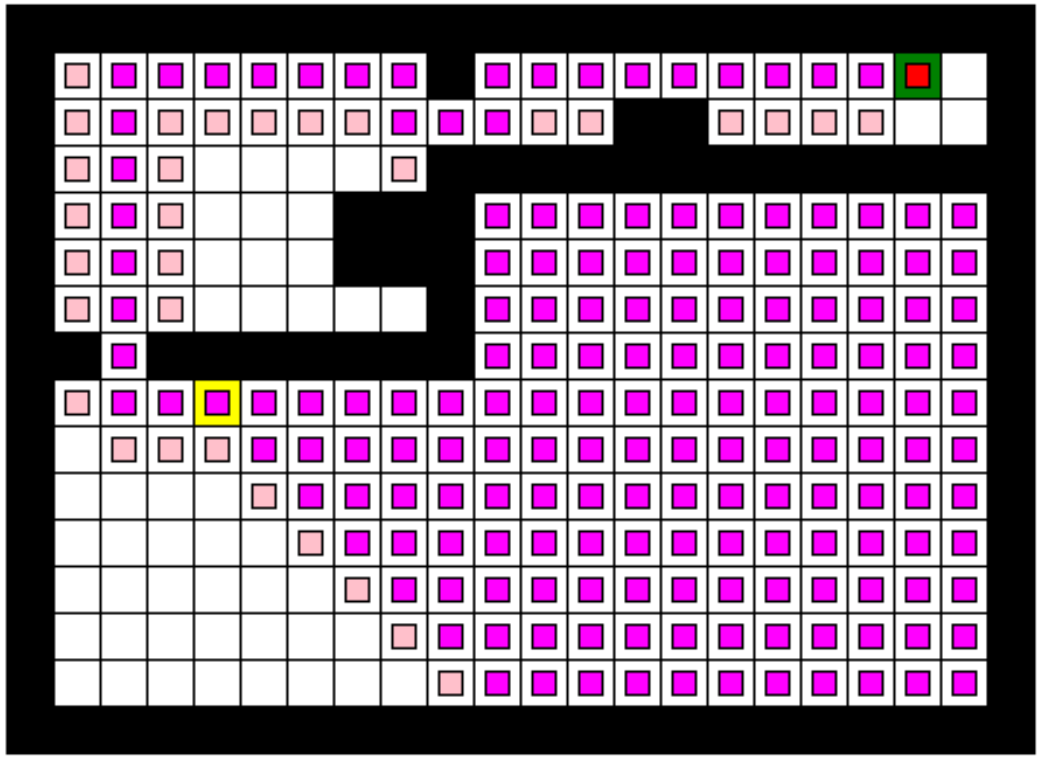
W dolnej części labiryntu heurystyka trochę prowadzi na manowce

A^* w labiryncie (3)



ale jeżeli w poprzednim labiryncie przebić drzwi, to wówczas znowu jest prawie idealnie.

A^* w labiryncie (4)



Heurystyka mocno „oszukana” przebiegiem labiryntu.

Koniec nagrania 1

Właściwości A^*

Twierdzenie 1

A^* jest zupełny (warunki jak dla UCS).

Twierdzenie 2

Jeżeli h jest spójna, to A^* zwraca optymalną ścieżkę (wersja grafowa)

Twierdzenie 3

Jeżeli h jest optymistyczna, to A^* w drzewie zwraca optymalną ścieżkę.

Dowody: będą, ale najpierw jeszcze trochę praktyki.

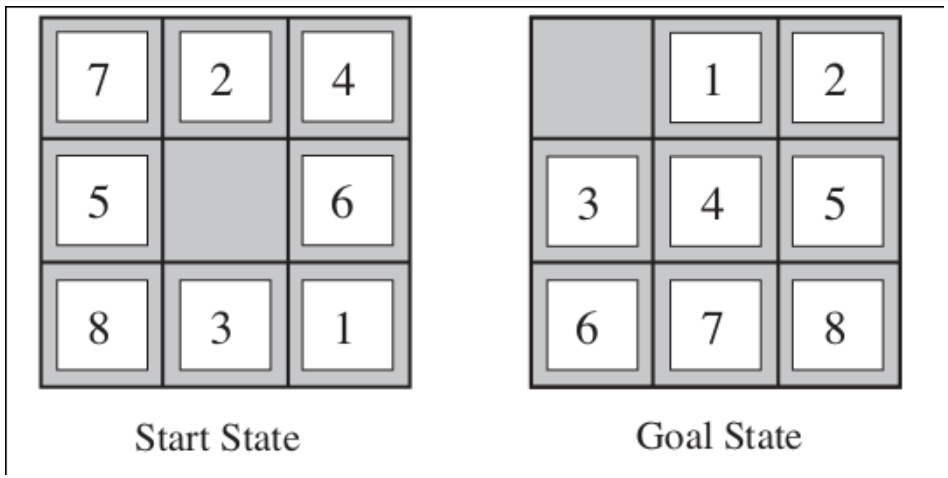
Drobna uwaga

$A*$ oczywiście nie daje gwarancji znalezienia rozwiązania dla grafów nieskończonych, w których istnieją nieskończone ścieżki o skończonej sumie wag krawędzi
Argument taki sam jak dla UCS

Heurystyki dla ósemki

Uwaga

Pewne aspekty tworzenia heurystyk można dość dobrze prześledzić na przykładzie ósemki



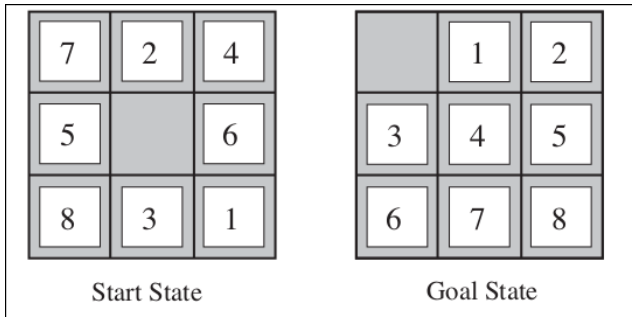
Pytanie

Jak (optymistycznie) oszacować odległość tych dwóch stanów?

- Heurystyka musi być prosta do policzenia.
- Przy projektowaniu heurystyki kluczowe jest pilnowanie **optymizmu** (czyli że niedoszacujemy odległości).
- Choć teoretycznie wymagany jest silniejszy warunek (spójności), ale w praktyce naturalne optymistyczne heurystyki są spójne...
- ... a o optymizm łatwiej zadbać (i łatwiej przypomnieć sobie definicję)

Zanim przewiniesz slajd spróbuj chwilę pomyśleć o optymistycznym szacowaniu liczby ruchów w ósemce

Heurystyki dla ósemki (2)



Pomysł 1

Jak coś jest nie na swoim miejscu, to musi się ruszyć o co najmniej 1 krok. Zliczajmy zatem, ile kafelków jest poza punktem docelowym ($h_1(s) = 8$)

Pomysł 2

Jak coś jest nie na swoim miejscu, to musi pokonać cały dystans do punktu docelowego. Zliczajmy zatem, ile kroków od celu jest każdy z kafelków ($h_2(s) = 3 + 1 + 2 + 2 + 2 + 3 + 3 + 2 = 18$)

Pytanie

Która intuicyjnie jest lepsza?

- Intuicja mówi, że jeżeli coś **dokładniej** szacujemy, to algorytm bazujący na tych dokładniejszych szacunkach będzie działał lepiej
- Z dwóch optymistycznych heurystyk ta, która daje większe wartości, jest dokładniejsza.

Efektywność w praktyce

- Dla h_2 efektywność A^* jest 50000 razy większa niż IDS.
- Istnieją heurystyki dające jeszcze 10000 krotne przyspieszenie dla 15-ki, a milionowe dla 24-ki (wobec h_2)

Relaksacja

Kiedy możliwy jest ruch w łamigłówce ósemka? Docelowe pole jest: (koniunkcja warunków):

a) sąsiadujące

b) wolne

Możemy rezygnować z części (lub wszystkich) warunków, otrzymując łatwiejsze łamigłówki.

Uwaga

Liczba ruchów w łatwiejszym zadaniu od startu do punktu docelowego jest często sensowną heurystyką w zadaniu orygialnym.

Relaksacja

Heurystyka h_1

Ruch możliwy jest **zawsze**.

Heurystyka h_2

Ruch możliwy jest **gdy pole jest obok** (niekoniecznie puste).

Heurystyka h_3

Ruch możliwy jest **gdy pole jest puste** (niekoniecznie obok).

Relaksacja na mapie

Relaksacja w zadaniu poszukiwania w labiryntach lub przy podróży samochodem drogami:

Relaksacja na mapie

Relaksacja w zadaniu poszukiwania w labiryntach lub przy podróży samochodem drogami:

1. W labiryncie: pomijanie ścian, czyli odległość taksówkową

Relaksacja na mapie

Relaksacja w zadaniu poszukiwania w labiryntach lub przy podróży samochodem drogami:

1. W labiryncie: pomijanie ścian, czyli odległość taksówkową
2. W atlasie drogowym: pomijanie dróg, czyli odległość euklidesową (helikopterem)

Operacja maksimum dla heurystyk

- Jak mamy dwie heurystyki h_1 i h_2, obie optymistyczne
- to możemy zdefiniować $h_3(x) = \max(h_1(x), h_2(x))$

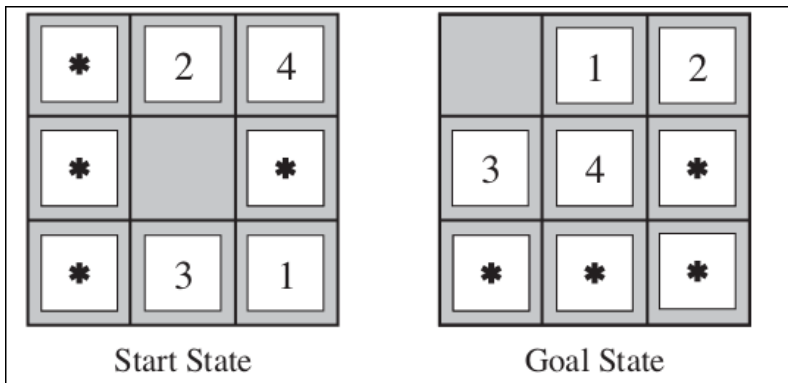
Uwaga

h_3 też będzie optymistyczna!

Bazy wzorców

Heurystyki możemy budować korzystając z baz wzorców, zapamiętujących koszty rozwiązań podproblemów danego zadania.

Przykład:



Działanie bazy wzorców

- Znajdujemy wszystkie podproblemy dla danego stanu (które mamy w bazie)
- A następnie bierzemy maksimum kosztów jako wartość heurystyki
- Możemy do tego maksimum dołożyć jakieś proste heurystyki (typu h_2).

Pytanie

A czy nie moglibyśmy użyć sumowania, zamiast maksimum?

Działanie bazy wzorców (2)

- Niestety suma daje niedopuszczalne heurystyki (bo pewne ruchy liczymy wielokrotnie, gwiazdki w jednym wzorcu są istotnymi kafelkami w innym)
- **Pytanie:** Jak temu zapobiec?
- **Odpowiedź:** stosując „rozłączne" wzorce (nic się nie powtarza) i w każdym wzorcu liczyć tylko ruchy kafelków z liczbami.
  To to są te najefektywniejsze heurystyki dla 8-ki

Uwaga

Niemniej warto wiedzieć, że czasem rezygnuje się z optymalności i stosuje niedopuszczalne heurystyki (które czasem przeszacowują odległość), ze względu na szybkość działania.

Koniec nagrania II (lub III)

Algorytm A*. Przypomnienie

Definicje

- $g(n)$ – koszt dotarcia do węzła n
- $h(n)$ – szacowany koszt dotarcia od n do (najbliższego) punktu docelowego ($h(s) \geq 0$)
- $f(n) = g(n) + h(n)$

Algorytm

Przeprowadź przeszukanie, wykorzystując $f(n)$ jako priorytet węzła (czyli rozwijamy węzły od tego, który ma najmniejszy f).

Własności A^*

Plan

Spróbujemy dowieść następujących rzeczy:

1) A^* zwraca najkrótszą drogę
2) A^* jest zupełny.

Dowód optymalności

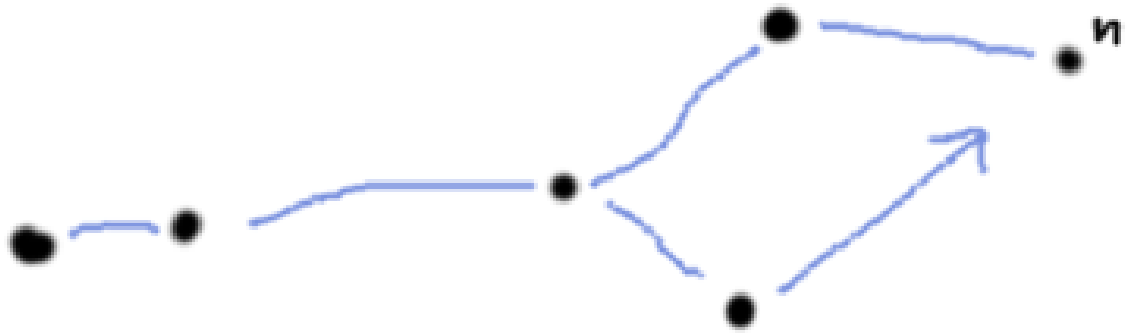
Potrzebujemy dwóch faktów:

1. (F1.) Jeżeli h jest spójna, wówczas na każdej ścieżce wartości f są niemalejące.
   D-d (n' jest następnikiem n):

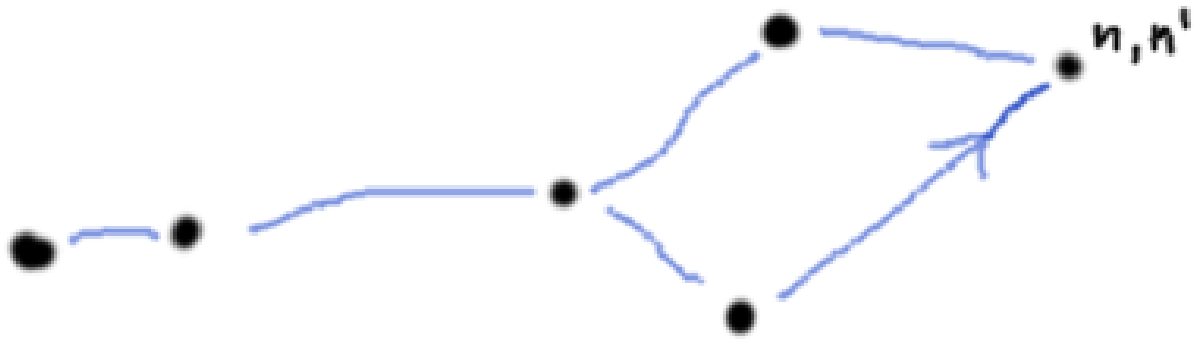
   $$f(n') = g(n') + h(n') = g(n) + c(n, n') + h(n') \geq g(n) + h(n) = f(n)$$

2. (F2.) Zawsze, gdy algorytm bierze węzeł do rozwinięcia, to koszt dotarcia do tego węzła jest optymalny (najmniejszy możliwy).

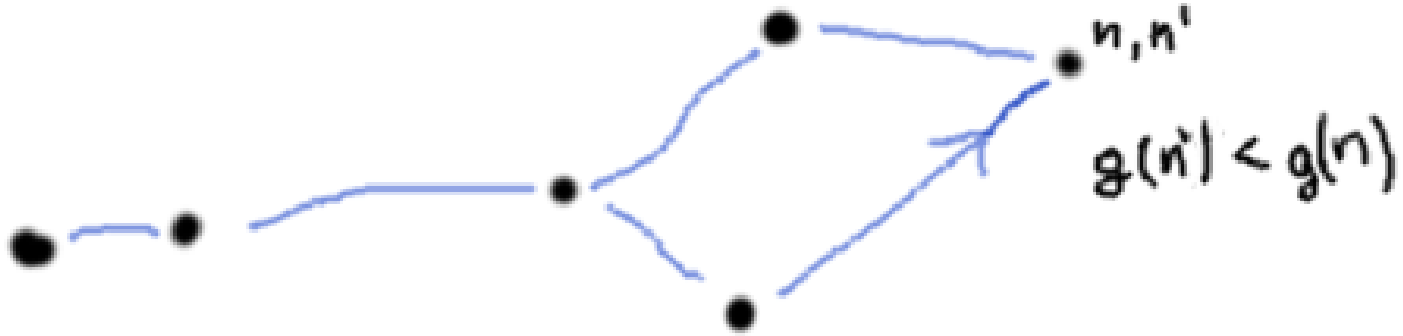
Dowód F2



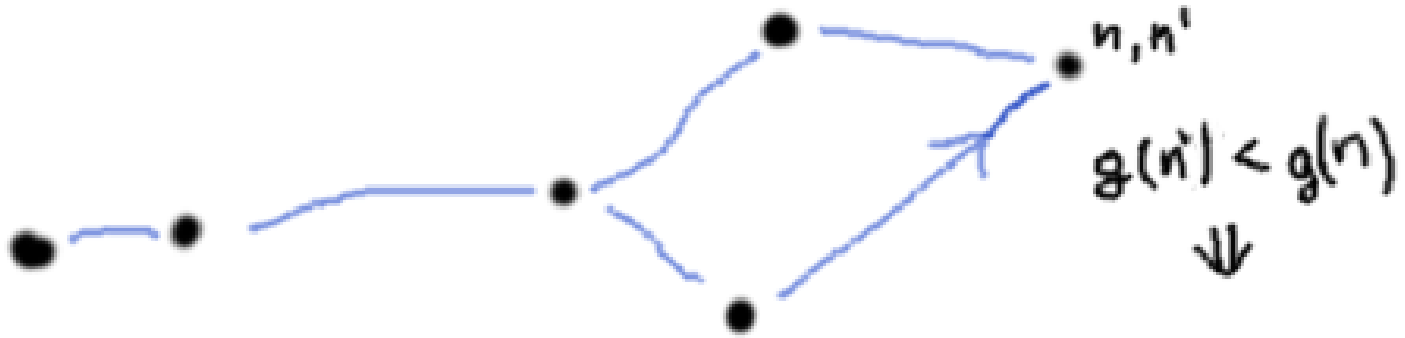
Dowód F2



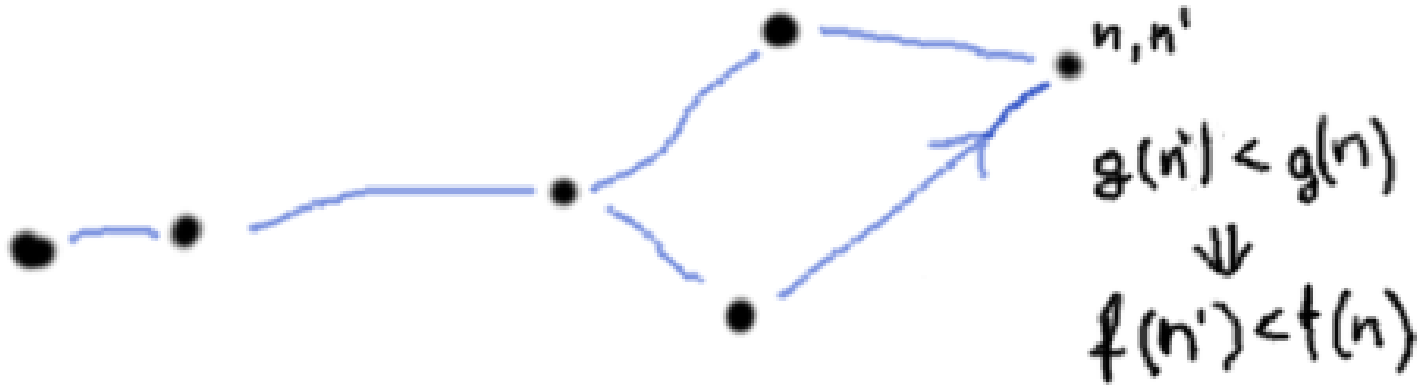
Dowód F2



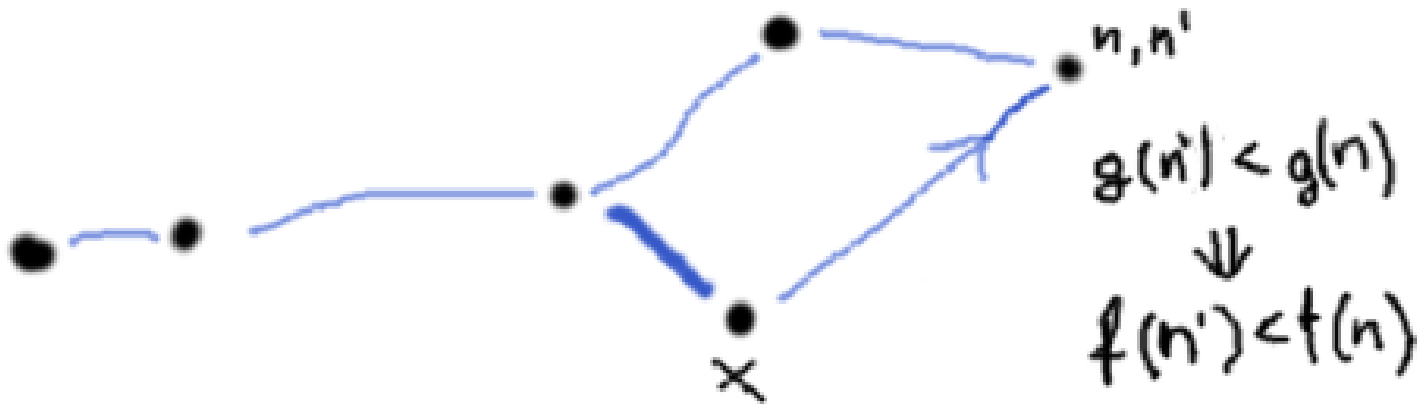
Dowód F2



Dowód F2

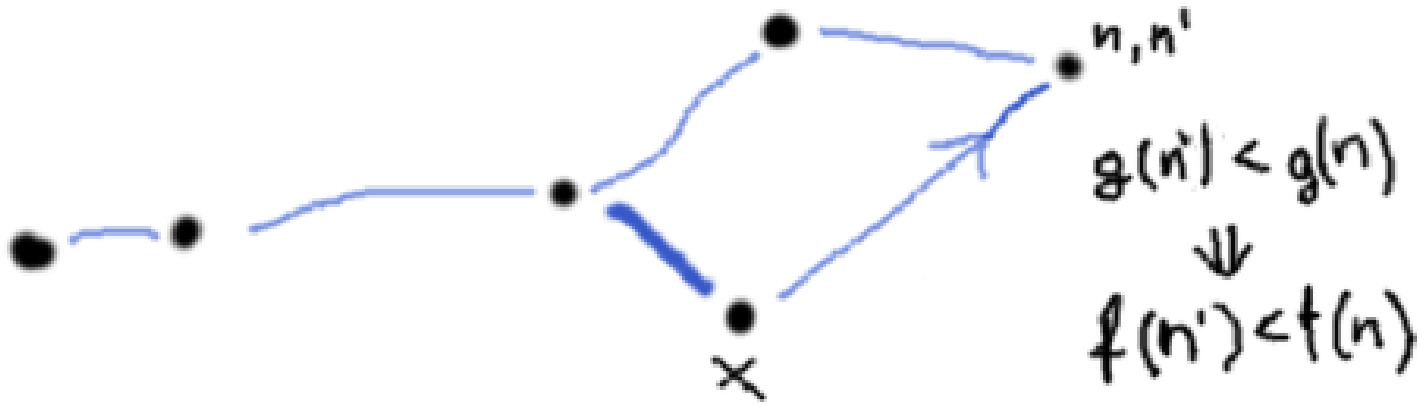


Dowód F2



z F1: $f(x) \leqslant f(n') < f(n)$

Dowód F2



z F1: $f(x) \leq f(n') < f(n)$

SPRZECZNOŚĆ

Dowód F2

- Bierzemy nie wprost węzeł n, do którego kolejne dotarcie daje mniejszy koszt niż dotarcie pierwsze.
- Wartość f dla tego węzła drugi raz jest mniejsza (h takie same, g mniejsze)
- Na ścieżce od początku do n' drugiego mamy widziany w pierwszym przebiegu węzeł x (tuż po rozgałęzieniu)
- Z własności F1 mamy: $f(x) < f(n)$

Zatem algorytm powinien wybrać x a nie n. **Sprzeczność.**

Optymalność

Popatrzmy na pierwszy znaleziony węzeł docelowy (n_{end})

- $f(n_{\text{end}}) = g(n_{\text{end}}) + h(n_{\text{end}}) = g(n_{\text{end}})$ (bo h jest rozsądna)
- Każdy kolejny węzeł docelowy jest nielepszy, bo dla niego $g(n) \geq g(n_{\text{end}})$

Zupełność

Niech C^* będzie kosztem najtańszego rozwiązania ($g(n_{\text{end}})$)

- Algorytm ogląda wszystkie węzły, dla których $f(n) < C^*$
- Być może oglądnie również pewne węzły z konturu $f(n) = C^*$, przed wybraniem docelowego n, t.że $f(n) = g(n) + 0 = C^*$

Uwaga

Skończona liczba węzłów o $g(n) \leq C^*$ gwarantuje to, że algorytm się skończy. Do skończoności z kolei wystarczy założyć, że istnieje $\varepsilon > 0$, t.że wszystkie koszty są od niego większe bądź równe.

Lepsze heurystyki

Uwaga

A^* nie rozwija węzłów t.że $f(n) > C^*$. Zatem im większa h (przy założeniu spełniania warunków dobrej heurystyki), tym lepsza.

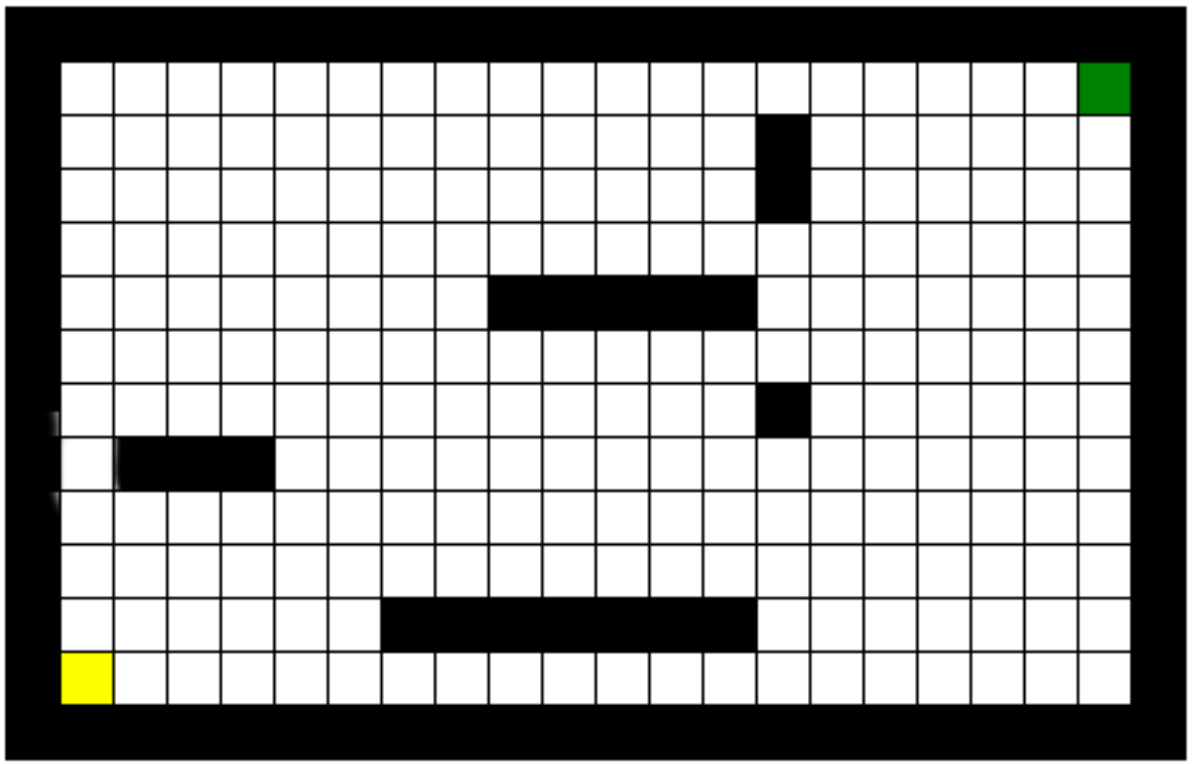
Konsekwencja

Mając dwie spójne (optymistyczne) heurystyki h_1 i h_2, możemy stworzyć $h_3(n) = \max(h_1(n), h_2(n))$, która będzie lepsza od swoich składników (szczegóły na ćwiczeniach).

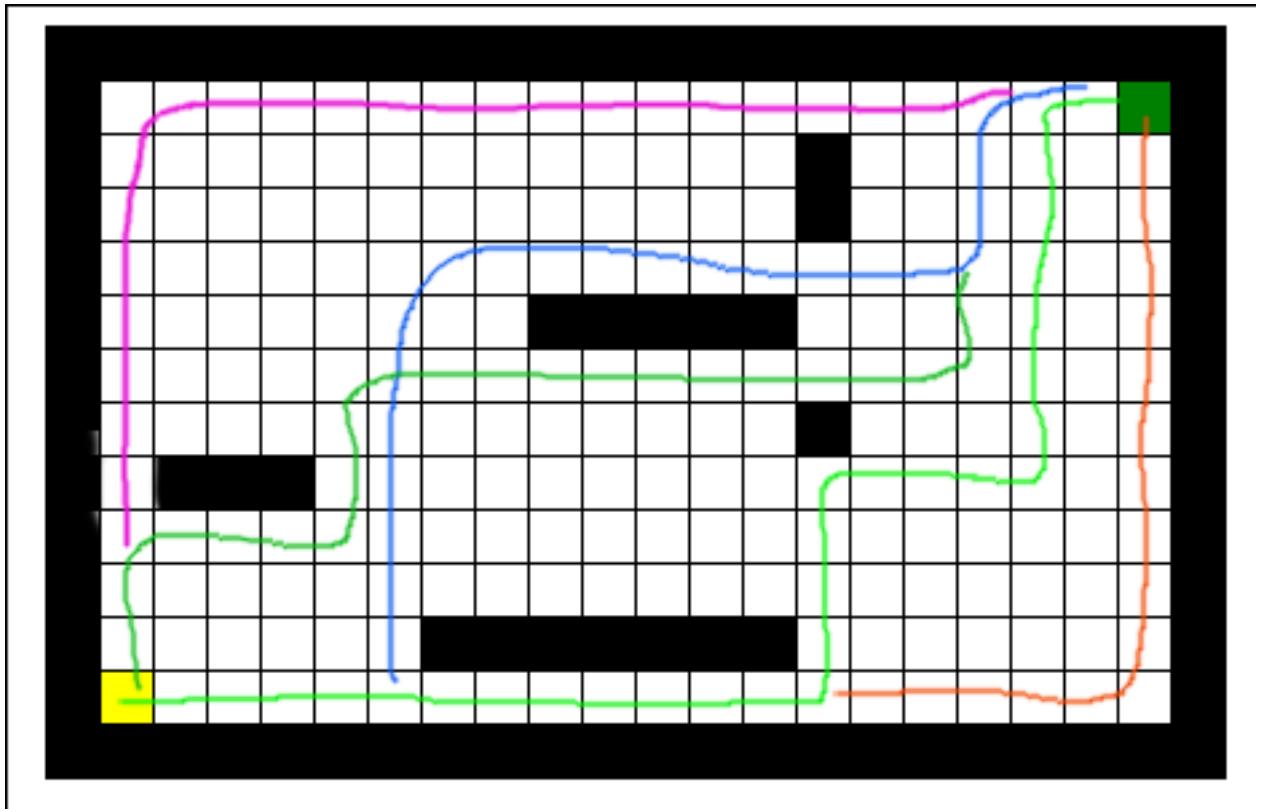
Pytanie

Co się będzie działo, jeżeli nasza funkcja h będzie liczyła dokładną odległość od celu?

Bierzemy heurystykę Manhatańską (przyjmijmy, że cel jest jeden), czyli $h(n) = |g_x - n_x| + |g_y - n_y|$



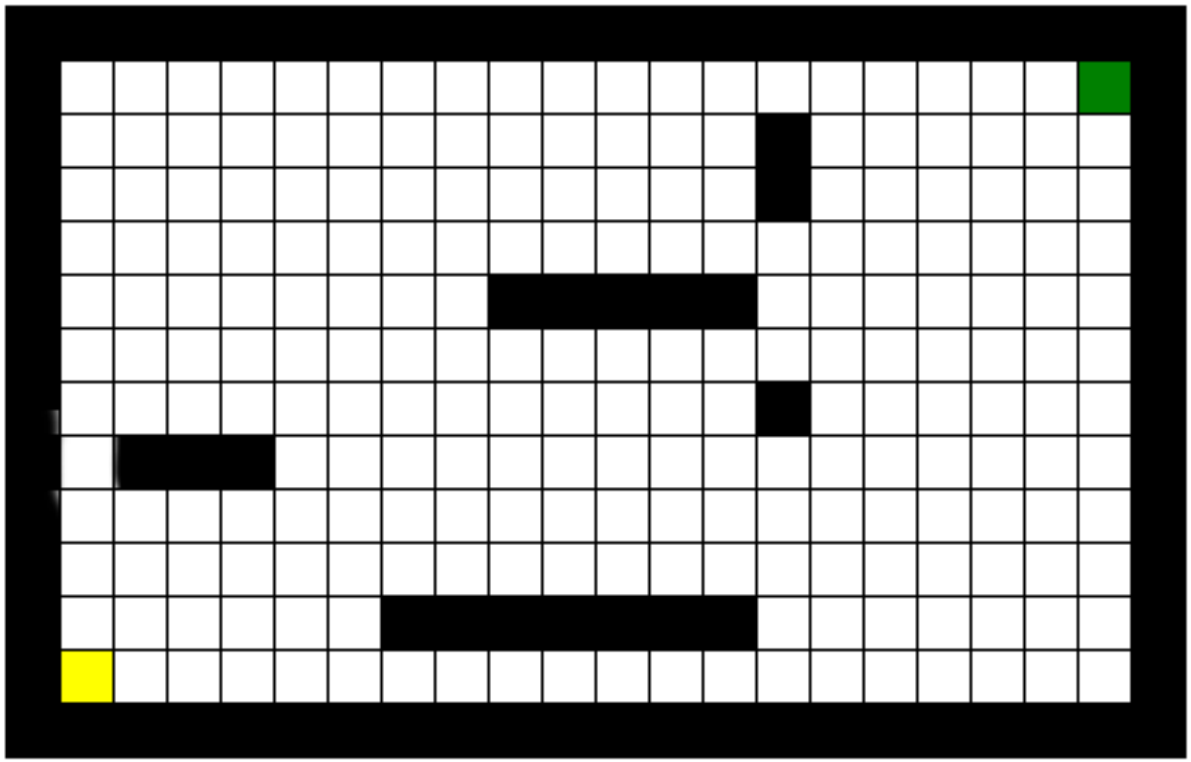
Płaska funkcja f



Wszystkie ścieżki idące w prawo i do góry są optymalne. Funkcja f jest stała.

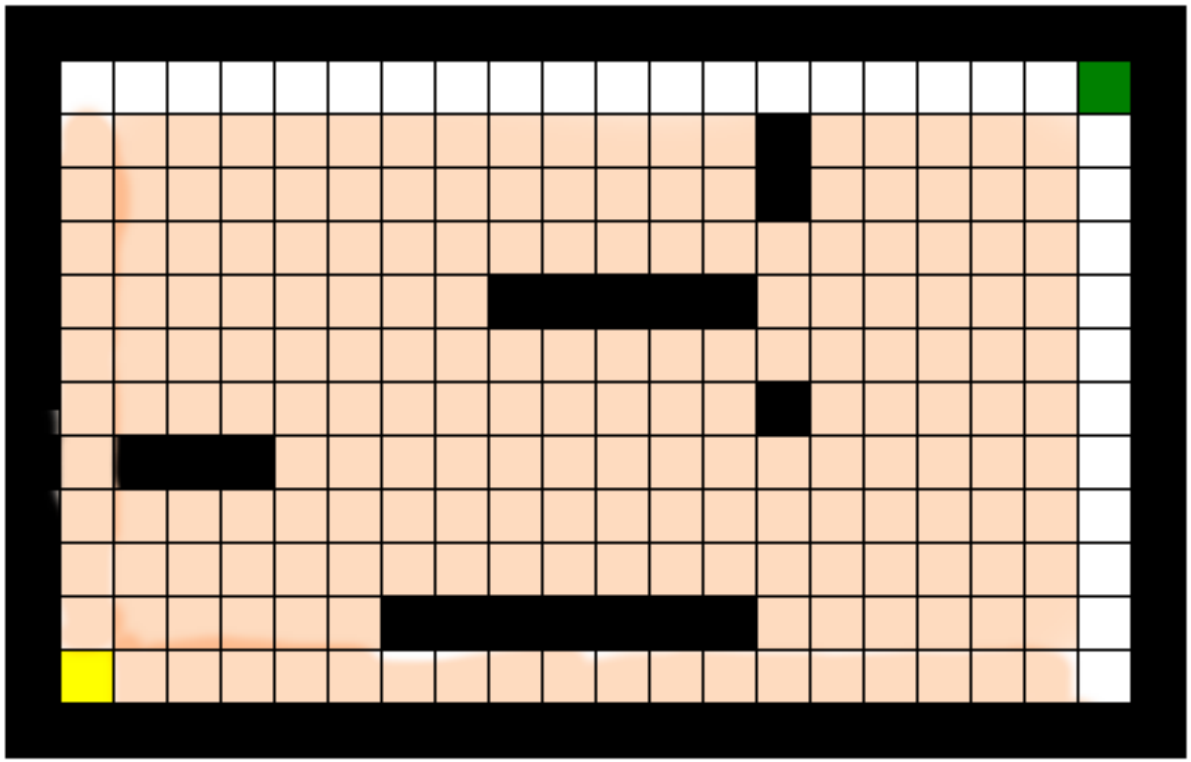
Porównanie heurystyk

Porównajmy na przykładzie heurystykę manhatańską i euklidesową. Która jest lepsza? **Poniżej zaznaczone węzły, które na pewno musi obejrzeć A^* z heurystyką Euklidesową**

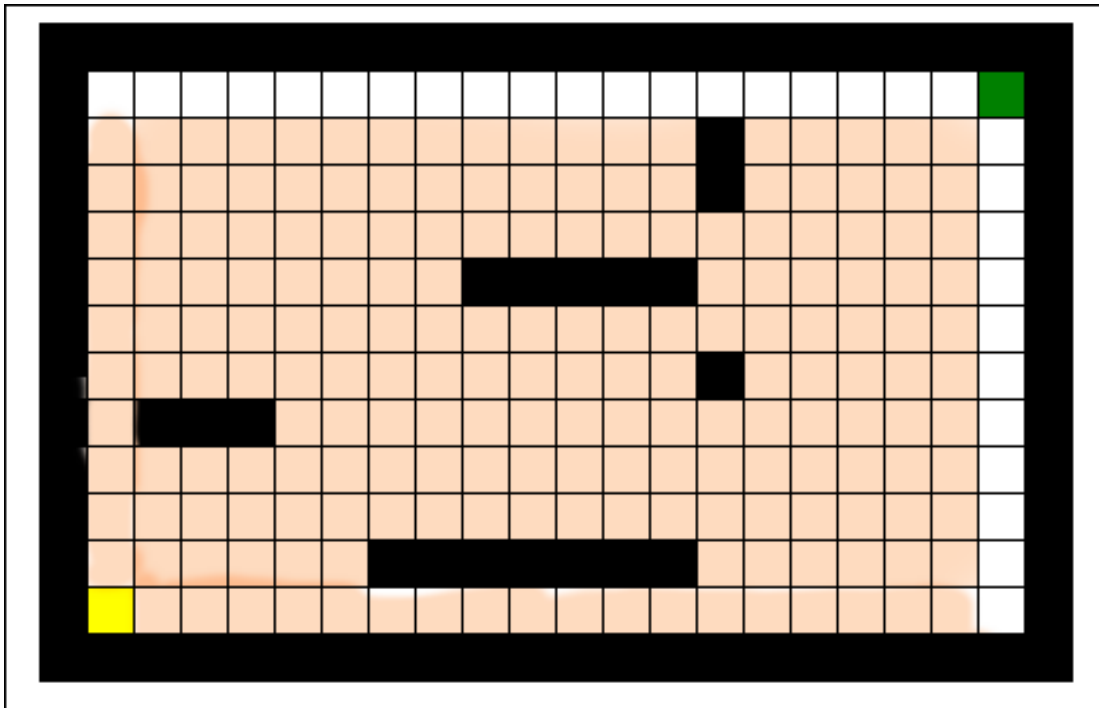


Porównanie heurystyk

Porównajmy na przykładzie heurystykę manhatańską i euklidesową. Która jest lepsza? **Poniżej zaznaczone węzły, które na pewno musi obejrzeć A^* z heurystyką Euklidesową**



Uzasadnienie



- Funkcja f z heurystyką h_M ma wszędzie wartość 30
- Funkcja f z heurystyką h_E osiąga wartość 30 jedynie na brzegach.