Politechnika Warszawska

MSI

5. Przeszukiwanie przestrzeni stanów

Włodzimierz Kasprzak

Układ

1. Schemat przeszukiwania przestrzeni stanów
2. Przeszukiwanie drzewa a przestrzeń stanów
3. Strategie ślepego przeszukiwania
4. Strategia „best-first” („najpierw najlepszy”)
5. Strategia poinformowana (z heurystyką)

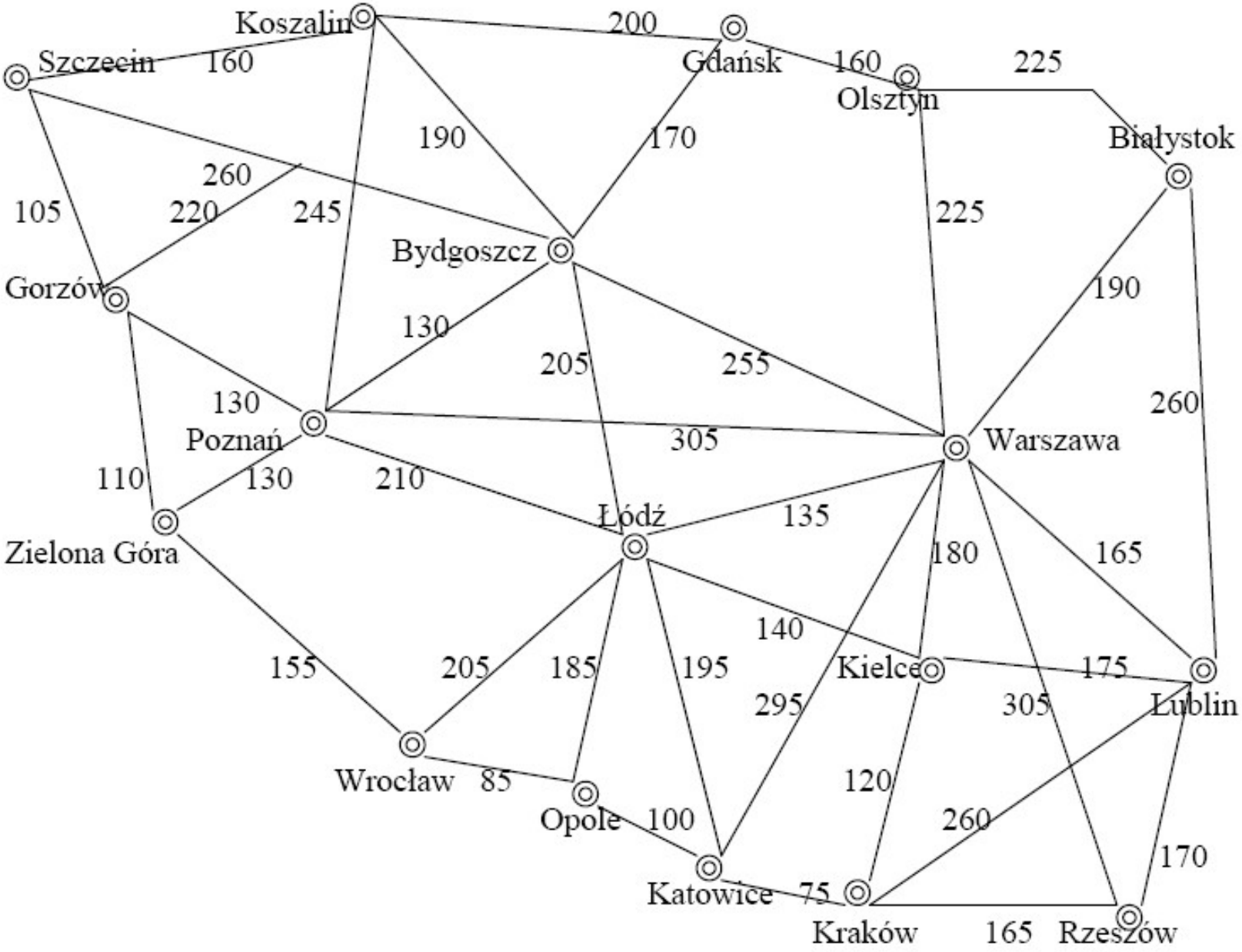
1. Schemat przeszukiwania przestrzeni stanów

Przykład: powrót ze Szczecina do Krakowa

- Aktualnie jesteśmy w Szczecinie. Chcemy wrócić do domu transportem drogowym.
- Sformułowanie celu:
  - Dotrzeć do Krakowa.
- Sformułowanie problemu (w postaci grafu stanów) :
  - Sytuacja agenta: przebywanie w danym mieście
  - Akcje (operatory): przejazd pomiędzy miastami
- Znalezione rozwiązanie:
  - Sekwencja miast wyznaczająca drogę przejazdu;

  Np. (Szczecin, Bydgoszcz, Łódź, Katowice, Kraków).

Przykład: graf stanów problemu



Definicja problemu przeszukiwania

Problem przeszukiwania definiowany jest przez 4 pojęcia:

1. Stan początkowy - np.: Szczecin = „*agent jest w Szczecinie*"
2. Akcje (operacje): $A = \{a_1, a_2, \ldots a_n\}$, i funkcja następnika :

   $[(stan, a) \rightarrow stan\_wy] \in S \times A \times S$

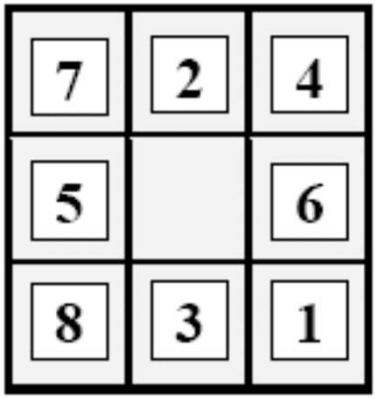
   Np.: (Szczecin, *ze Szczecina do Bydgoszczy*) → *Bydgoszcz*
3. Warunek osiągnięcia celu – spełniony w stanie końcowym:
   - jawny (*explicit*), np., stan T = „*w Krakowie*"
   - niejawny (*implicit*), np., *Szachmat*(T) = true
4. Koszt rozwiązania (addytywny koszt ścieżki)
   - np., suma odległości, liczba wykonanych akcji, itp.
   - $c(x, a, y)$ jest kosztem akcji i powinno być $c \geq 0$

Rozwiązaniem jest sekwencja akcji (ścieżka) prowadząca od stanu początkowego do końcowego.

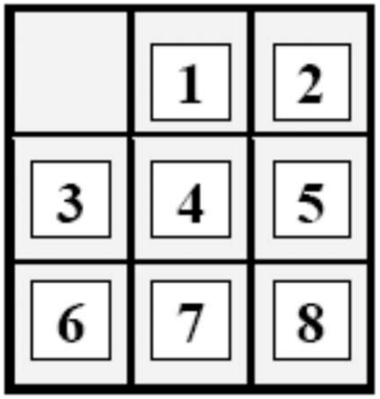
Wybór przestrzeni reprezentacji problemu

- Świat rzeczywisty jest bardzo złożony
  → trzeba stworzyć jego model uproszczony (abstrakcyjny)
- (Abstrakcyjny) stan odpowiada wielu sytuacjom rzeczywistym
- (Abstrakcyjna) akcja odpowiada wielu rzeczywistym akcjom
  - np., „Szczecin → Bydgoszcz" reprezentuje zbiór możliwych dróg, objazdów, miejsc odpoczynku, etc.
- (Abstrakcyjne) rozwiązanie
  - Odpowiada zbiorowi rzeczywistych dróg, które w rzeczywistym świecie prowadzą do celu
- Każda z abstrakcyjnych akcji powinna stanowić uproszczenie oryginalnej czynności

Przykład: 8-puzzli



Stan początkowy



Stan końcowy

- stany - położenie płytek (wartości dyskretne)
- akcje – przemieszczenie pustego miejsca w: lewo, prawo, górę, dół
- warunek stopu – podany stan końcowy
- koszt akcji - 1 za każdy ruch

[Uwaga: optymalne rozwiązanie *n*-puzzli jest NP-trudne]

Przeszukiwanie drzewa

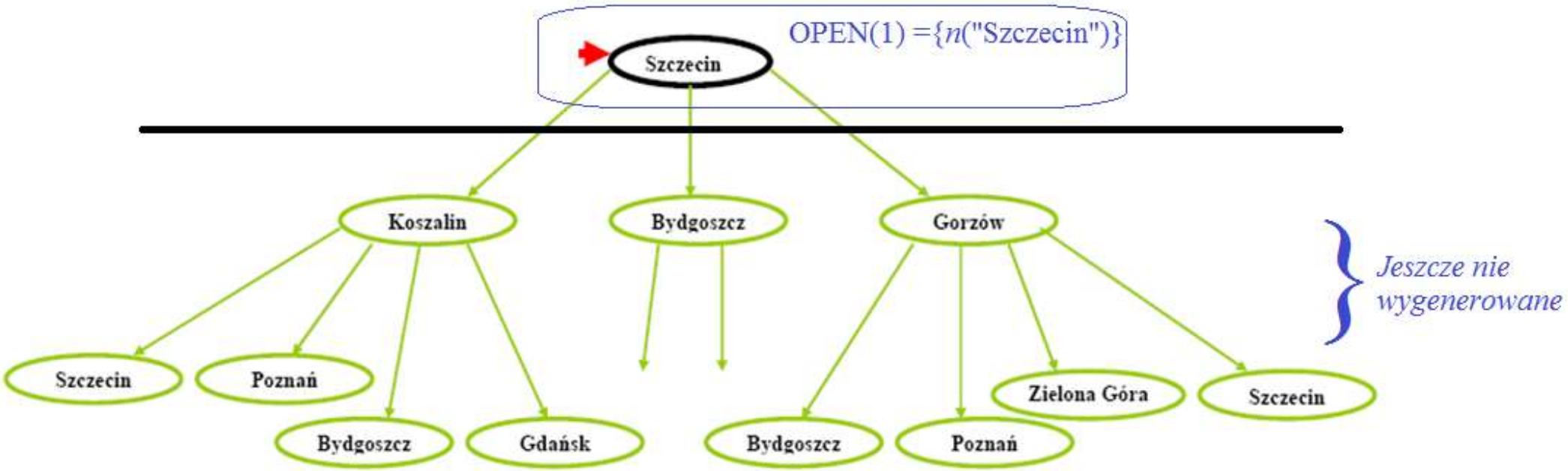
Podstawowe zasady:

- realizujemy eksplorację przestrzeni stanów problemu metodą generowania następników już zbadanych *sytuacji* (reprezentowanych węzłami drzewa);
- rozwijamy drzewo przeszukiwania od korzenia, reprezentującego *sytuację początkową*, poprzez węzły pośrednie do jednego z liści drzewa, reprezentującego *sytuację końcową* – spełniającego *warunek zatrzymania* („stopu”);
- w problemie „jedno-stanowym” *sytuacja* odpowiada pojedynczemu stanowi problemu; w problemie „wielo-stanowym” jest to w ogólności grupa stanów;
- aktualny zbiór węzłów-liści gotowych do rozwinięcia tworzy tzw. *skraj* drzewa (uporządkowana lista OPEN);
- sposób uporządkowania węzłów w skraju wyraża określoną *strategię przeszukiwania*.

Drzewo decyzyjne (przeszukiwania)

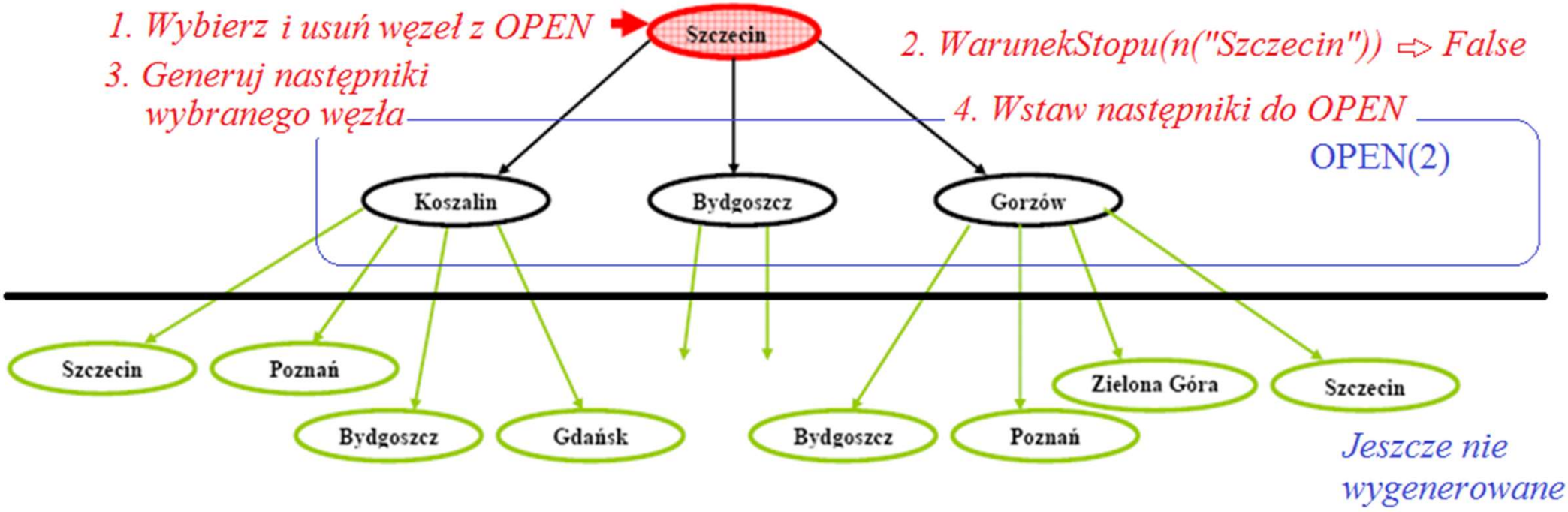
Przykład

Początkowe drzewo przeszukiwania (decyzyjne) dla problemu „powrotu do Krakowa”. Jedyny węzeł początkowy *n*(„Szczecin”) reprezentuje sytuację „*agent jest w Szczecinie*” (jest to także jeden stan problemu). Stanowi on *skraj* (OPEN) dla pierwszej iteracji.



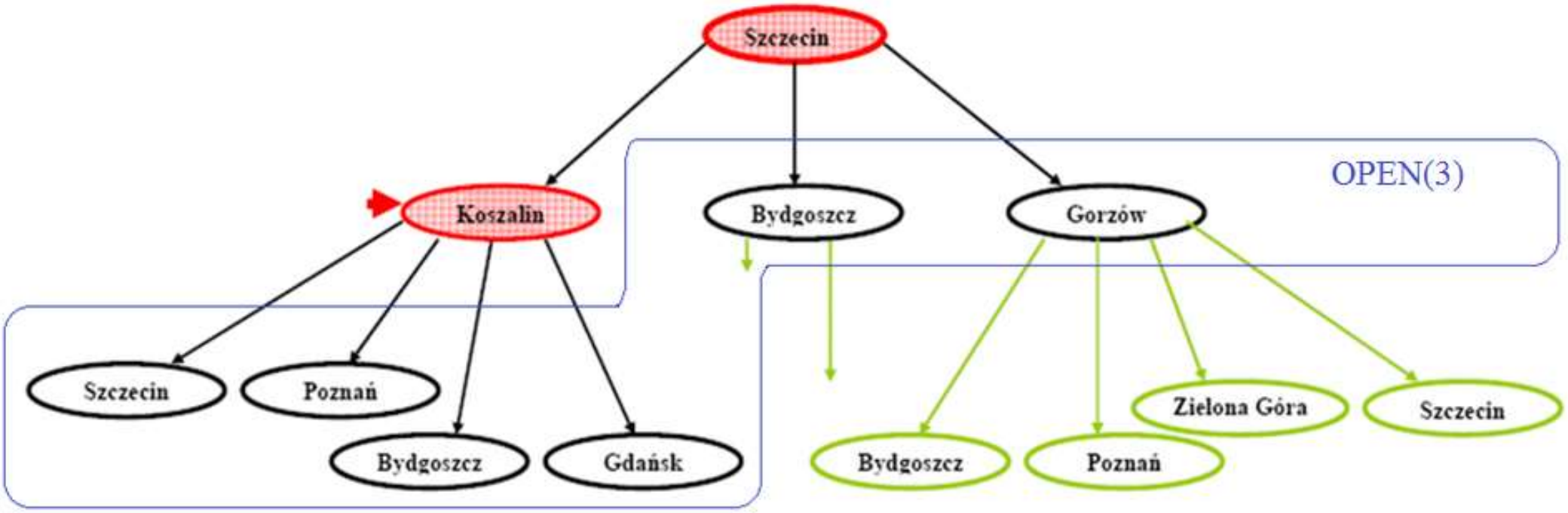
Przykład – drzewo decyzyjne (1)

(1) Wybierany jest węzeł „Szczecin”, sprawdzany jest na nim warunek stopu (zatrzymania) („nie zachodzi”) i generowane są jego możliwe następniki – węzły dla stanów: „Koszalin”, „Bydgoszcz”, „Gorzów”. Węzły te stanowią skraj dla drugiej iteracji algorytmu.



Przykład – drzewo decyzyjne (2)

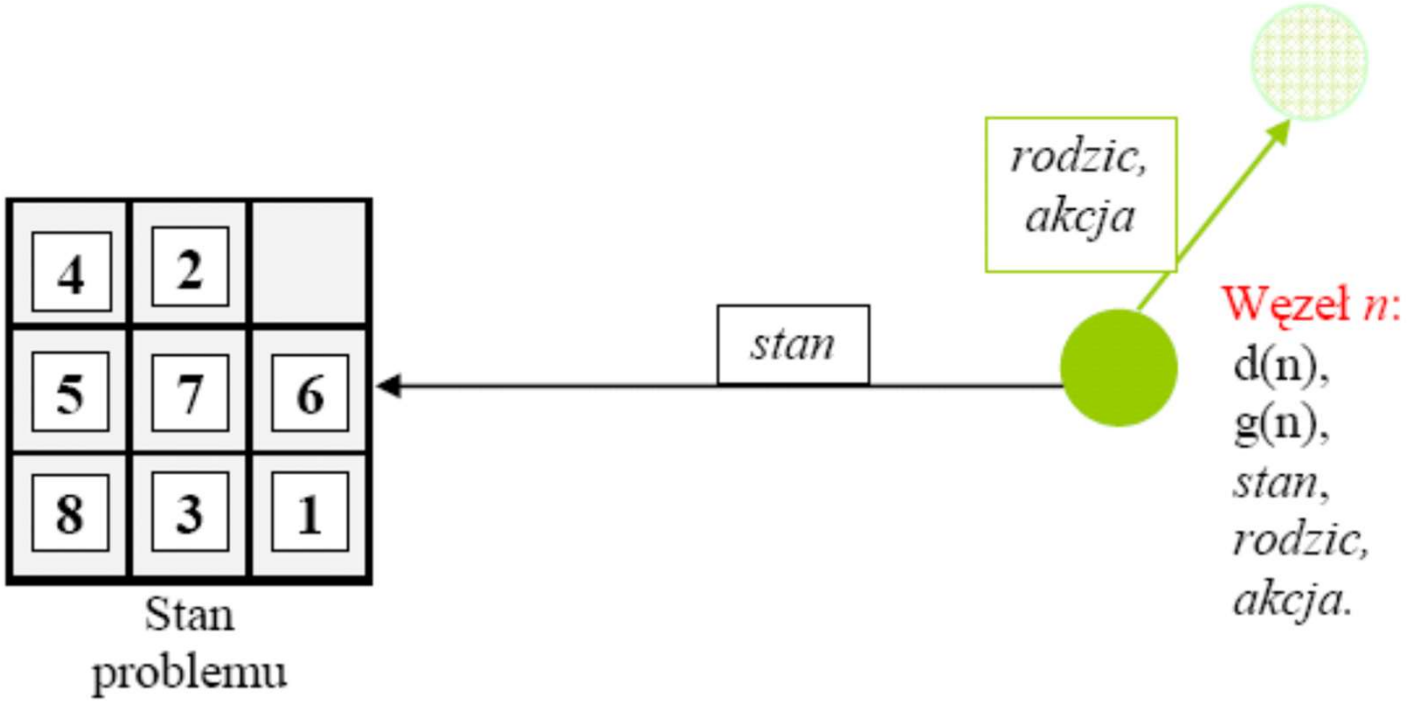
(2) Kolejny wybierany węzeł to np. *n*(„Koszalin”). Nie spełnia on warunku celu („stopu”) więc generowane są z kolei jego następniki.



(3) itd. ...

Stany problemu a węzły drzewa przeszukiwania (decyzyjnego)

- Stan jest reprezentacją fizycznej konfiguracji świata.
- Węzeł *n* jest strukturą danych, tworzącą część drzewa przeszukiwania, zawierającą: stan środowiska, wskaźnik do rodzica, akcję, koszt sekwencji akcji prowadzącej do tego węzła *g*(*n*) i głębokość węzła d(*n*).



Kryteria oceny strategii przeszukiwania

- Kryteria oceny strategii przeszukiwania:
  - zupełność: jeżeli istnieje rozwiązanie, to czy będzie ono znalezione?
  - złożoność czasowa: całkowita liczba węzłów generowanych w czasie przeszukiwania,
  - złożoność pamięciowa: maksymalna liczba węzłów jednocześnie rezydujących w pamięci,
  - optymalność: czy zawsze znajdowane jest rozwiązanie najlepsze?
- Złożoność czasowa i pamięciowa są wyrażane poprzez parametry:

  b: maksymalne rozgałęzienia drzewa przeszukiwania,

  d: głębokość, na której znajduje się najtańsze rozwiązanie,

  m: maksymalna głębokość drzewa przeszukiwania.

2. Przeszukiwanie drzewa a grafu

- Strategia przeszukiwania drzewa jest wyznaczona poprzez wybór kolejności rozwijania (wizytowania) węzłów, znajdujących się w skraju drzewa (na tzw. liście OPEN).
- Strategia przeszukiwania grafu powinna uwzględniać fakt, że mogą być wielokrotnie generowane węzły „równoważne”, tzn. reprezentujące te same sytuacje problemu (np. ten sam stan problemu). Bez dodatkowych kroków algorytmu przeszukiwania może to prowadzić do jego zapętlenia się na nieskończonej ścieżce.
- W szczególności w strategiach przeszukiwania grafu wprowadzana jest obok OPEN (liście drzewa przeszukiwania wymagające rozwinięcia) także druga **lista CLOSED**, na której umieszczane są węzły już poprzednio wizytowane (węzły niekońcowe lub już sprawdzane liście).

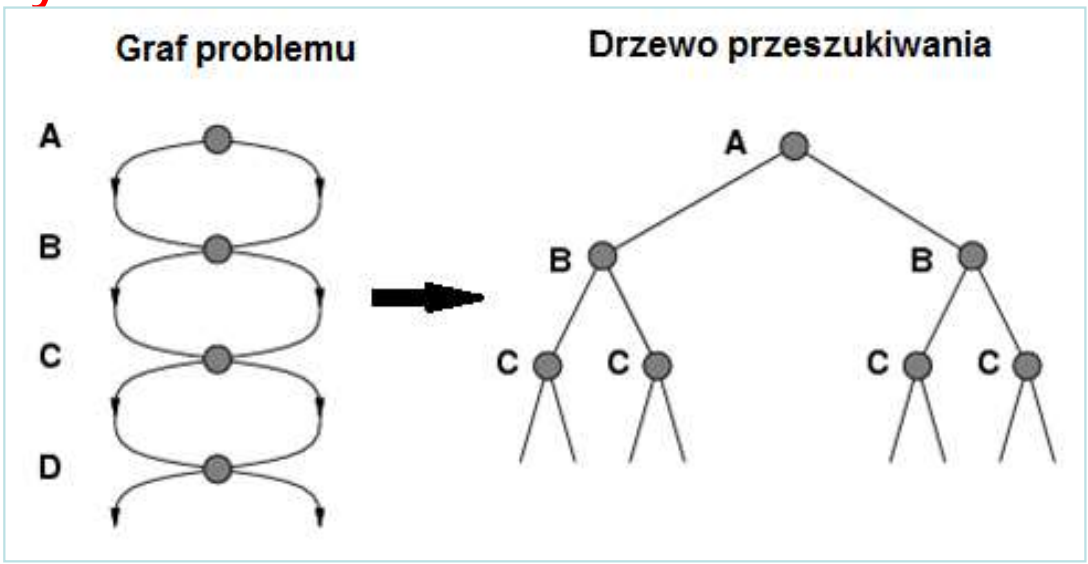
Unikanie powtarzania równoważnych węzłów drzewa przeszukiwania

- Wymagana jest taka modyfikacja strategii przeszukiwania, aby nadal zarządzała ona „drzewem przeszukiwania”, ale w przypadku wygenerowania węzła równoważnego z już istniejącym (w OPEN lub CLOSED) zastępowała nim poprzedni węzeł tylko wtedy, gdy w ocenie tej strategii prowadzi on do „lepszego” (w sensie kryteriów tej strategii) rozwiązania problemu. W przeciwnym przypadku nowo generowany węzeł nie jest dodawany do drzewa przeszukiwania.
- Przypadki generowania równoważnych węzłów drzewa (w sensie reprezentowania tych samych stanów problemu):

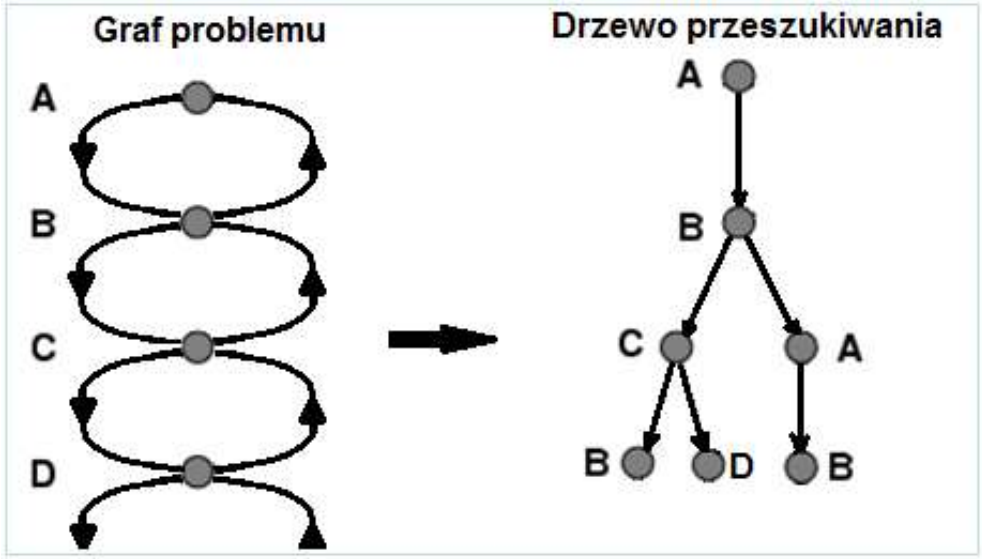
1. równoważne węzły znajdują się na różnych ścieżkach - może prowadzić do niepotrzebnego rozwijania nadmiarowych ścieżek i wpływać negatywnie na efektywność strategii przeszukiwania;
2. równoważne węzły są na tej samej ścieżce drzewa - odpowiada powstaniu pętli na ścieżce rozwiązania, co może prowadzić do ścieżki o nieskończonej długości i braku rozwiązania.

Powtarzanie węzłów

- Wielokrotne węzły w drzewie – różne ścieżki drzewa przeszukiwania:



- Powtórny węzeł na tej samej ścieżce (pętla):



3. Ślepe strategie przeszukiwania

Ślepe przeszukiwanie (*uninformed search*) wykorzystuje jedynie informację zawartą w sformułowaniu problemu.

Wyróżnimy następujące strategie ślepego przeszukiwania:

- Przeszukiwanie wszerz (*breadth-first search*)
- Przeszukiwanie z jednolitą funkcją kosztu (*uniform-cost search*)
- Przeszukiwanie w głąb (*depth-first search*)
- Przeszukiwanie z ograniczoną głębokością DLS (*depth-limited search*)
- Iteracyjne pogłębianie IDS (*iterative deepening search*).

Ślepe strategie przeszukiwania

1. Przeszukiwanie wszerz : skraj jest kolejką FIFO.
2. Przeszukiwanie z jednolitą funkcją kosztu: węzły uporządkowane są w skraju według niemalejących sumarycznych kosztów dotychczasowych akcji prowadzących do danego węzła
3. Przeszukiwanie w głąb: skraj jest stosem LIFO.
4. Przeszukiwanie z ograniczoną głębokością (DLS): tak jak w głąb do zadanego ograniczenia l, węzły na poziomie ograniczenia nie mają następników.
5. Iteracyjne pogłębianie (IDS): kolejne, niezależne od siebie wykonywanie przeszukiwań DLS dla coraz większych wartości ograniczeń (l=0,1,2,...) do momentu znalezienia celu.

Przeszukiwanie wszerz

W tej strategii przeszukiwania drzewa rozwijany jest zawsze najpłytszy dotąd nie rozwinięty węzeł.

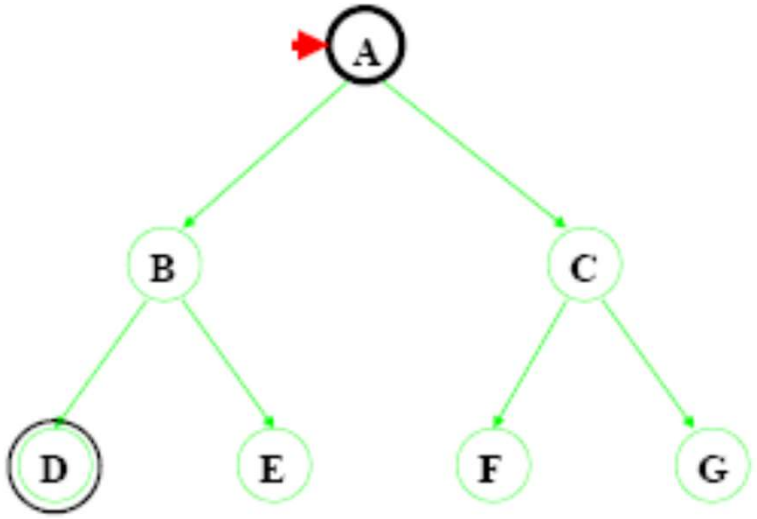
Implementacja strategii przeszukiwania wszerz polega na reprezentowaniu aktualnego *skraju drzewa* (tzw. zbiór OPEN) w postaci **kolejki FIFO** – nowo dodawane węzły-następniki ustawiane są zawsze na końcu kolejki a pobieranie węzłów (w celu rozwinięcia) ma miejsce na początku kolejki.

Przykład.

Dane jest drzewo przeszukiwania:

Węzeł A jest węzłem początkowym, a węzeł D – końcowym.

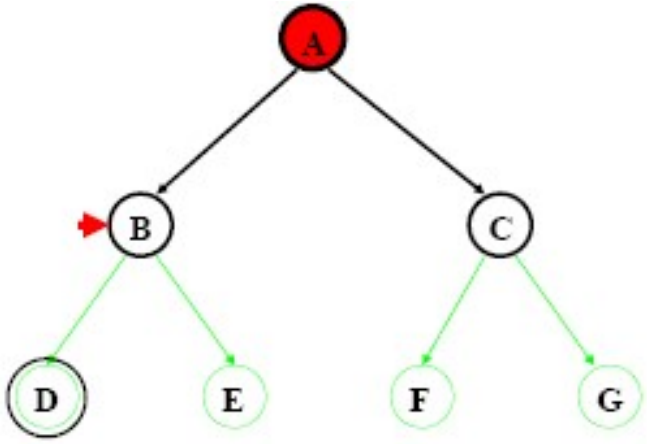
Kolejność rozwijanych węzłów w strategii przeszukiwania wszerz podano na następnej stronie.



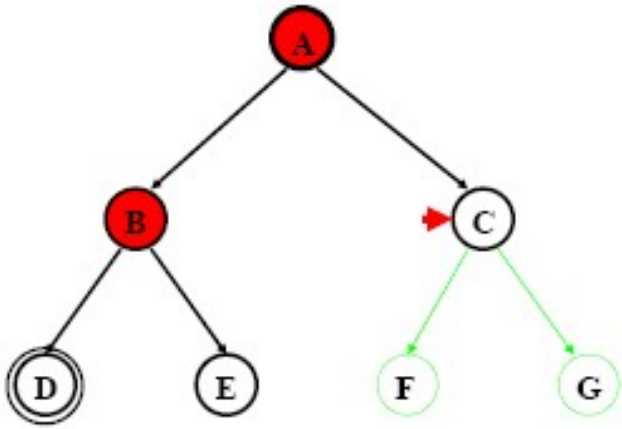
Przeszukiwanie wszerz - przykład

Iteracja 1: Skraj(1) = {A}, wybierany jest węzeł A, generowane są jego następniki B, C.

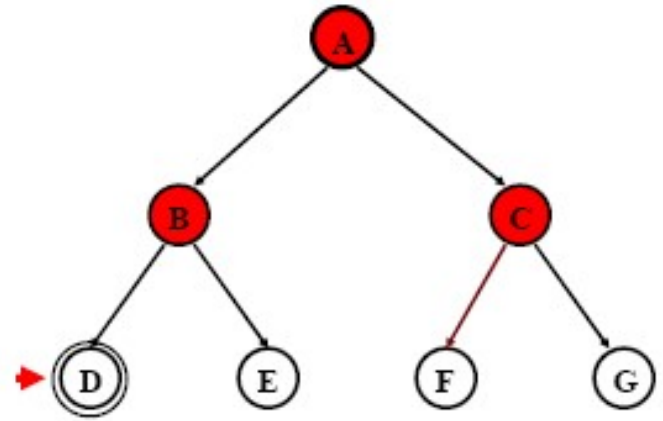
Iteracja 2: Skraj(2) = {B, C}, wybierany jest węzeł B, generowane są jego następniki D i E.



Iteracja 3: Skraj(3) = {C, D, E}, wybierany jest węzeł C, generowane są jego następniki F, G



Iteracja 4: Skraj(4) = { D,E,F,G}, wybierany jest węzeł D, nie są generowane żadne następniki, jeśli D spełnia warunek stopu to KONIEC.



Podsumowanie: rozwijane węzły A, B, C, D.

Przeszukiwanie z jednolitą funkcją kosztu („*uniform-cost" search*)

Zasada **przeszukiwania z jednolitą funkcją kosztu**: zawsze rozwija dotąd nie rozwinięty węzeł o najniższym koszcie z dotychczasowych.

Implementacja: *skraj* jest kolejką uporządkowaną według kosztu ścieżki (sekwencji akcji) prowadzącej do danego węzła.

Równoważne przeszukiwaniu wszerz, jeżeli koszty wszystkich akcji są równe.

Własności. Zupełność: tak, jeżeli koszt każdej akcji ≥ ε, gdzie $\varepsilon \geq 0$.

- Czas: liczba węzłów o koszcie $g(n)$ ≤ koszt optymalnego rozwiązania, $O(b^{(C^*/\varepsilon)})$, gdzie C^* to koszt optymalnego rozwiązania.
- Pamięć: liczba węzłów o koszcie $g(n)$ ≤ koszt optymalnego rozwiązania: $O(b^{(C^*/\varepsilon)})$.
- Optymalność: tak, w sensie minimalizacji kosztu, gdyż węzły rozwijane są zawsze w kolejności zwiększającego się kosztu $g(n)$.

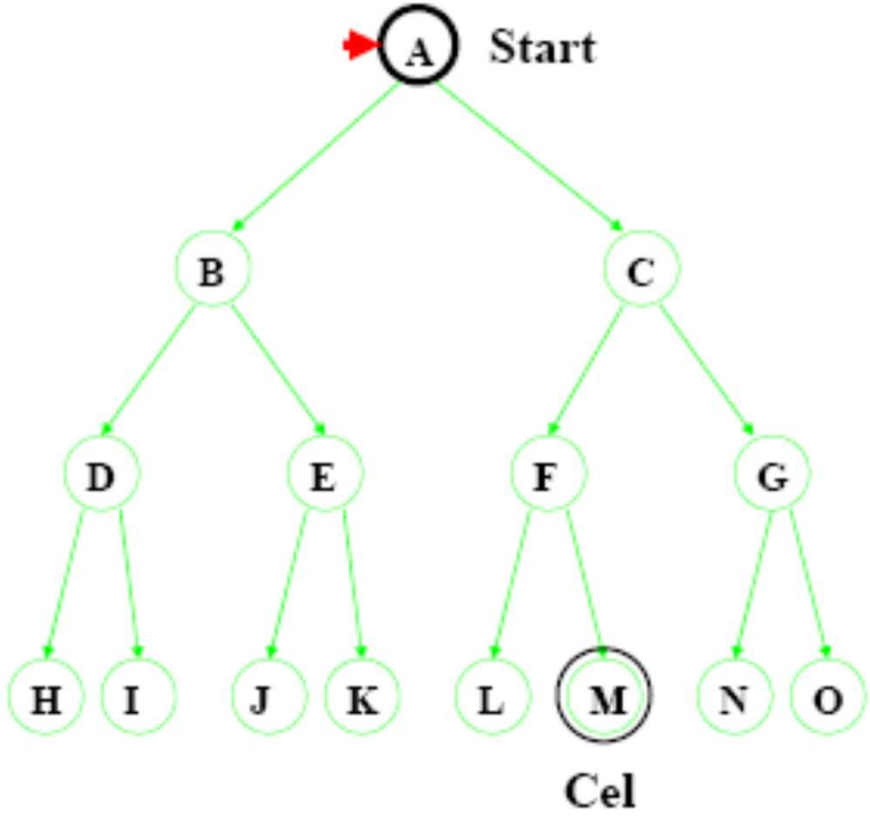
Przeszukiwanie w głąb

Zasada strategii przeszukiwania w głąb: rozwijany jest najgłębszy, dotąd nie rozwinięty węzeł

Implementacja *skraju drzewa* (listy OPEN) w postaci stosu LIFO; nowe następniki ustawiane są na początku stosu i są rozwijane w pierwszej kolejności. Taka kolejność rozwijania węzłów odpowiada, np. lewostronnemu obejściu drzewa.

Przykład:

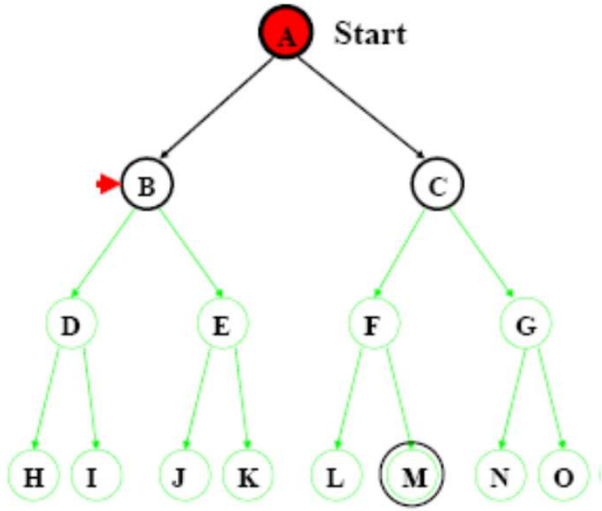
Dane jest drzewo przeszukiwania

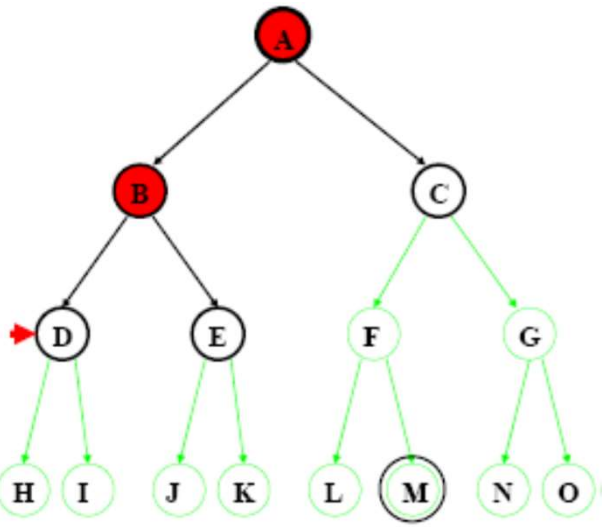


Przeszukiwanie w głąb - przykład

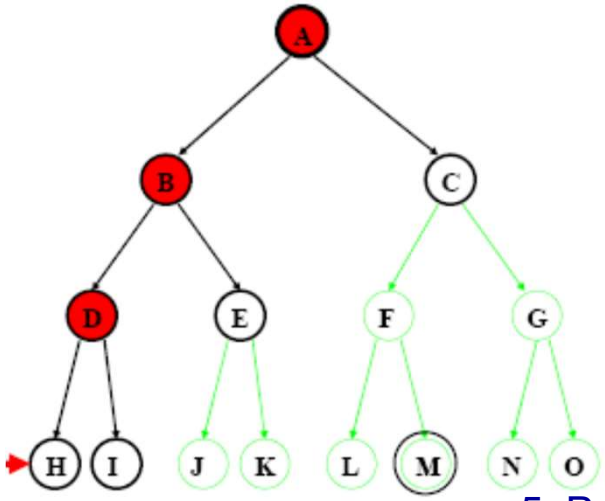
Krok 1: Skraj(1)={A}, wybór A.

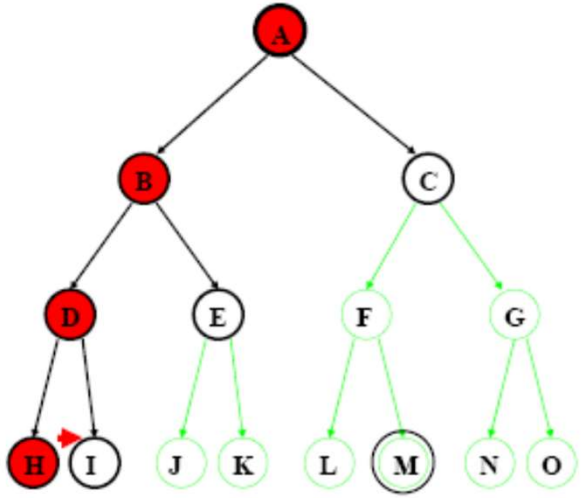
Krok 2: Skraj(2)={B,C}, wybór B. →Krok 3: Skraj(3)={D,E,C}, wybór D.





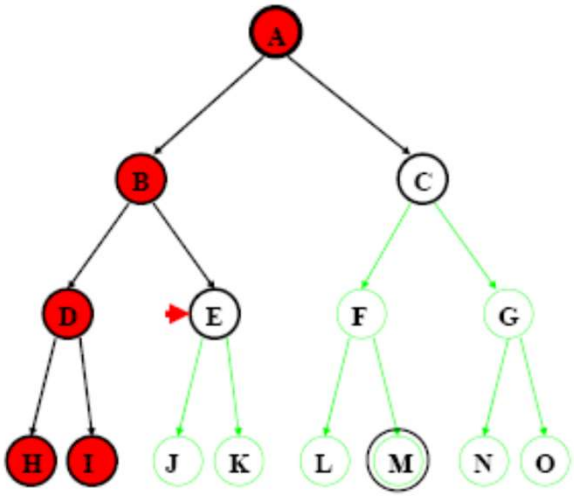
Krok 4: Skraj(4)={H,I,E,C}, wybór H. → Krok 5: skraj(5)={I,E,C}, wybór I.

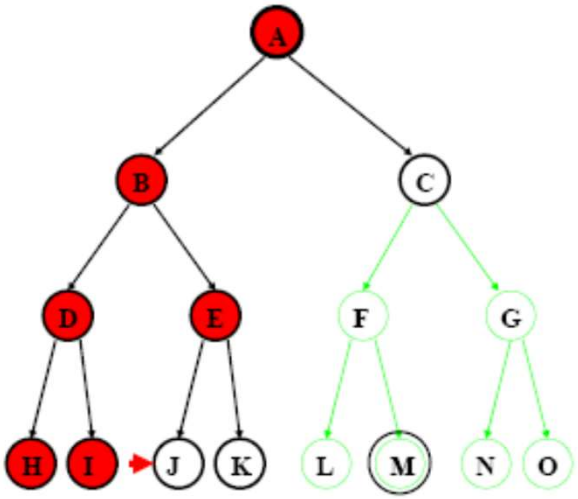




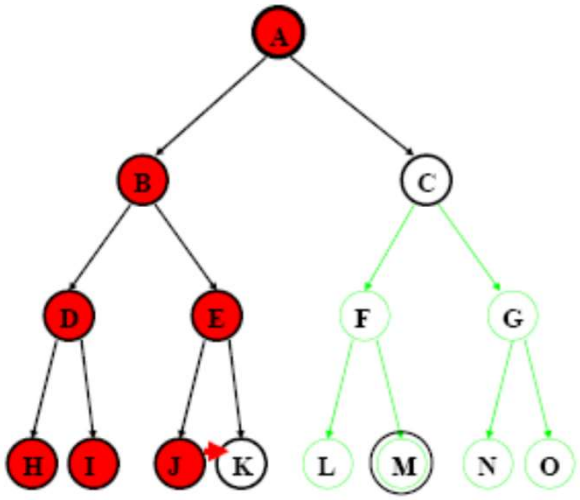
Przeszukiwanie w głąb – przykład (c.d.)

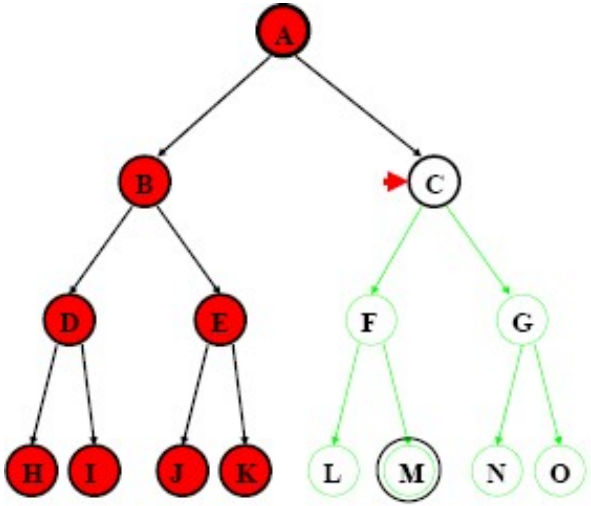
Krok 6 → Krok 7





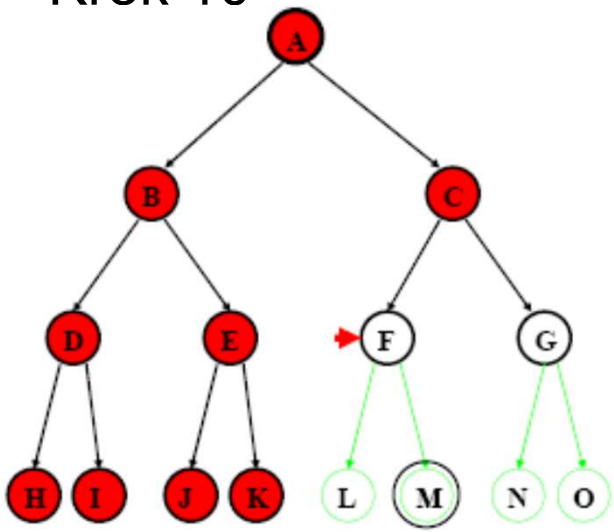
Krok 8 → Krok 9

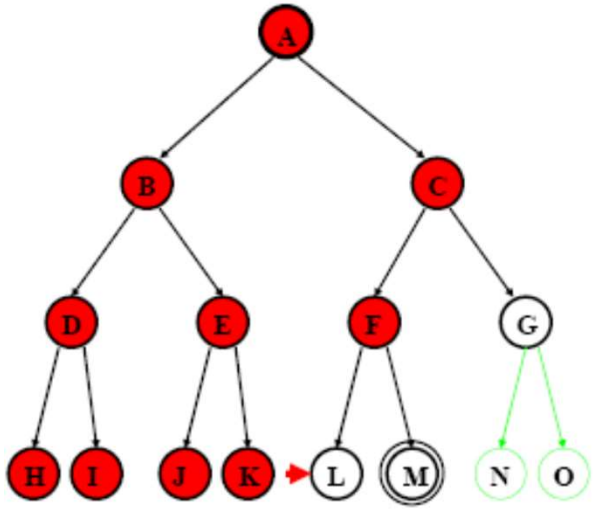




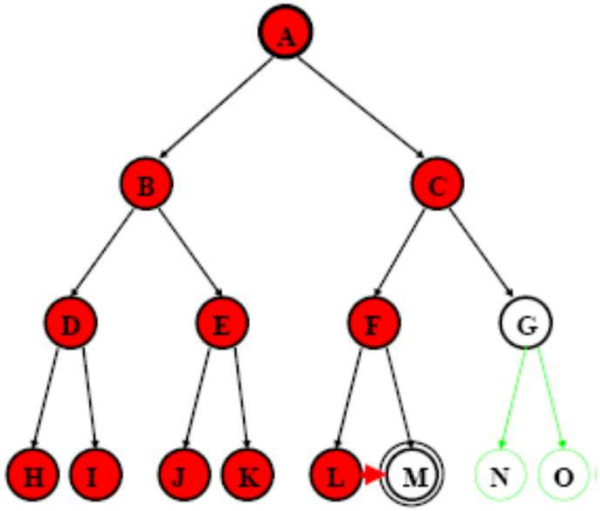
Przeszukiwanie w głąb – przykład (c.d.)

Krok 10 → Krok 11





Krok 12. Wybór M. Węzeł M spełnia warunek stopu (cel osiągnięty) → Koniec



Przeszukiwanie z ograniczoną głębokością (DLS)

Zasada: jest to odmiana przeszukiwania w głąb z ograniczeniem nałożonym na głębokość węzła, co oznacza, że jeśli l jest ograniczeniem głębokości węzła to (z punktu widzenia strategii) węzły na głębokości l nie posiadają następników.

Strategia nie jest ani zupełna ani optymalna. Jeśli każde rozwiązanie jest dane na głębokości większej niż l to nie zostanie znalezione żadne rozwiązanie (przeszukiwanie niezupełne). Przeciwnie, jeśli istnieją rozwiązania na ścieżkach krótszych niż l to nie ma gwarancji, że znalezione rozwiązanie jest optymalne, nawet w sensie minimalnej długości ścieżki. Jest to efektem stosowania strategii przeszukiwania w głąb w ramach ograniczenia do głębokości l (nieoptymalna).

Iteracyjne pogłębianie (IDS)

Zasada: iteracyjnie wywoływane jest przeszukiwanie z ograniczoną głębokością, od $l = 0$ do $l = \infty$, dopóki nie zostanie znalezione rozwiązanie, czyli ścieżka prowadząca do węzła końcowego (spełniającego warunek stopu).

Funkcja implementująca przeszukiwanie IDS:

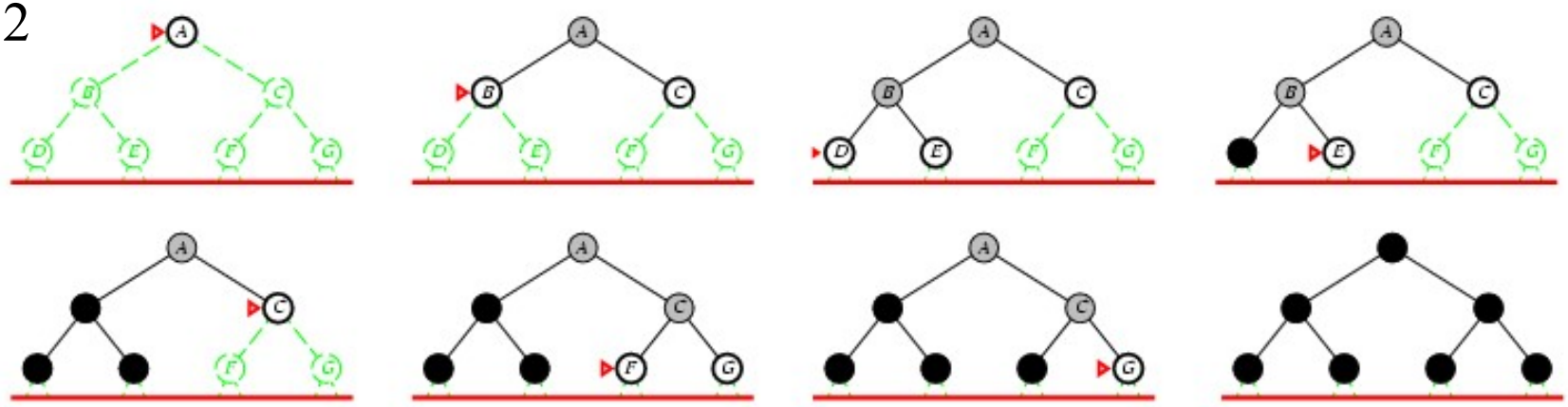
```
function Iteracyjne_pogłębianie(problem), Wynik: ścieżka
{  for (int głębia=0; głębia < ∞ ; głębia ++) {
        wynik ←Przeszukiwanie_z_ograniczoną_głębokością(
                                problem, głębia)
      if (Typ(wynik) = Ścieżka_końcowa) then return wynik;
    }
}
```

Iteracyjne pogłębianie - przykład

$l = 0$

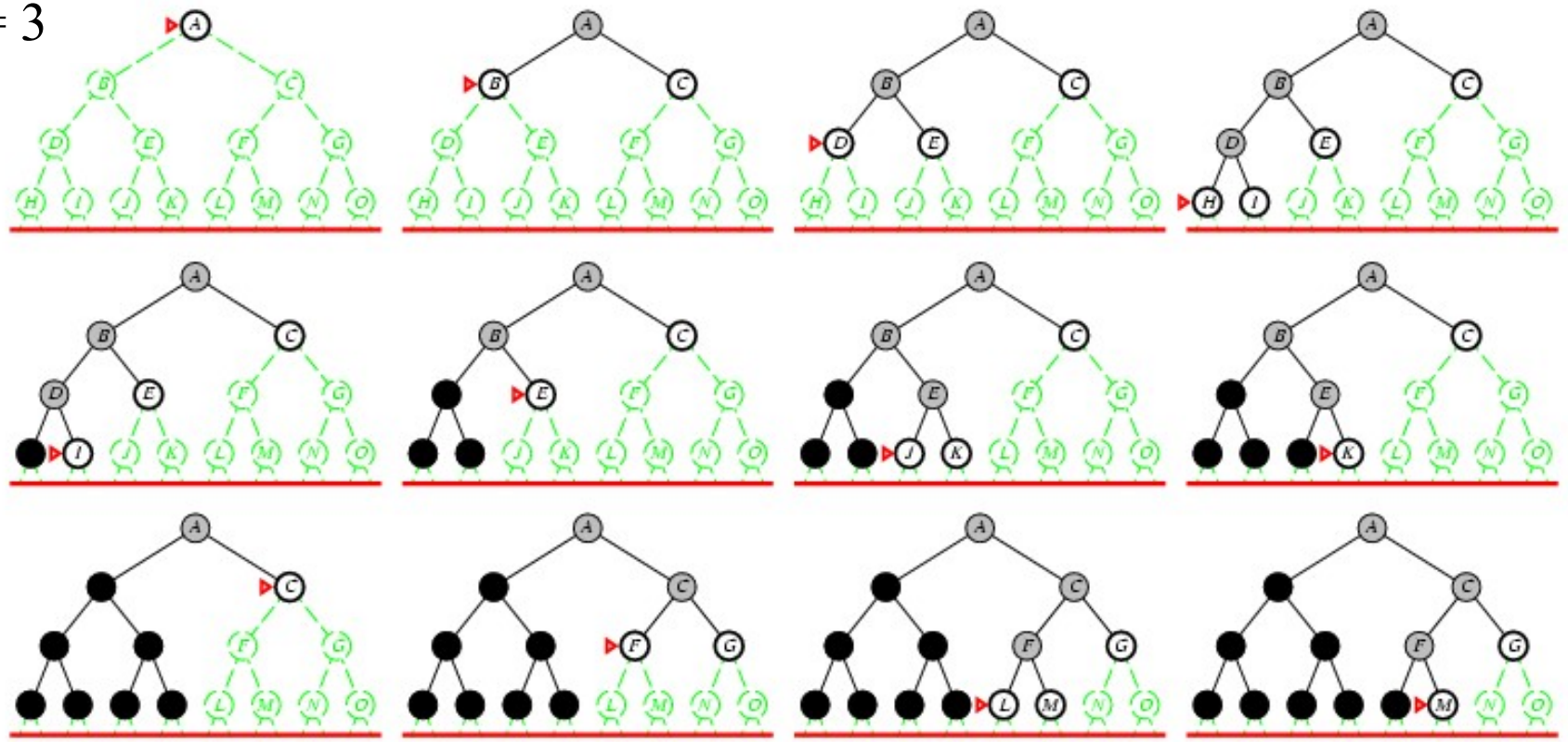
$l = 1$

$l = 2$



Iteracyjne pogłębianie – ostatnia iteracja

$l = 3$



Porównanie strategii ślepego przeszukiwania

Oznaczenia: b – stopień rozgałęzienia, d – długość ścieżki rozwiązania, m – maksymalna głębokość drzewa, l – *ograniczenie głębokości drzewa*, C^* - koszt optymalnego rozwiązania, ε - najmniejszy koszt akcji.

Strategia: / **Kryterium**	„wszerz”	„z jednorodnym kosztem”	„w głąb”	„z ograniczoną głębokością”	„z iteracyjnym pogłębianiem”
Zupełny ?	Tak	Tak	Nie	Nie	Tak
Złożoność czasowa	$O(b^{d+1})$	$O(b^{[C^*/\varepsilon]})$	$O(b^m)$	$O(b^l)$	$O(b^d)$
Złożoność pamięciowa	$O(b^{d+1})$	$O(b^{[C^*/\varepsilon]})$	$O(bm)$	$O(bl)$	$O(bd)$
Optymalny ?	Tak	Tak	Nie	Nie	Tak

Iteracyjne pogłębianie (IDS) znajduje optymalną ścieżkę, posiada liniową złożoność pamięciową i nie potrzebuje dużo więcej czasu niż inne ślepe algorytmy przeszukiwania drzewa. Dlatego stanowi dogodną alternatywę dla **przeszukiwania z jednorodnym kosztem**.

4. Strategia „best-first” (najpierw najlepszy)

- **Strategia przeszukiwania** jest wyznaczona sposobem wyboru kolejnych węzłów w drzewie (lub grafie) przeszukiwania tworzonym dla rozwiązania zadanego problemu.
- Strategia „best-first” (najpierw najlepszy) stosuje funkcję oceny $f(n)$ dla każdego węzła n, jednak ocena ta dotyczy jedynie dotychczasowych kosztów ścieżki: $f(n) = g(\text{start->}\ldots\text{->}n)$
- Strategia ”poinformowana” w ocenie węzła n uwzględnia to na ile „obiecujący” z punktu widzenia **celu** jest dany węzeł i wybiera (rozwija) taki najbardziej „obiecujący” węzeł.
- Implementacja obu strategii:
  - uporządkować węzły w skraju według oceny węzła $f(n)$, tzn. węzeł „najlepszy” jest umieszczany jako pierwszy, węzeł „drugi najlepszy” jest umieszczany po pierwszym, itd.
  - ustalić dodatkowe kryterium (porządek) dla węzłów o identycznej ocenie.

Klasyfikacja strategii „best first”

Strategia ślepa „jednorodnego kosztu”

- Strategia jednorodnego kosztu (wzgl. zysku), nazywana też strategią **równomiernego** kosztu (zysku) została już omówiona jako odmiana ślepego przeszukiwania. Posługuje się ona funkcją oceny węzła *n*, w której uwzględnia się jedynie koszty dotychczasowych akcji (na ścieżce od węzła początkowego do węzła *n*).
- Nie zalicza się jej do strategii poinformowanego przeszukiwania. Przeszukiwanie poinformowane ma miejsce wtedy, gdy posługujemy się w funkcji oceny oszacowaniem kosztów resztkowych $h(n)$ (nazywanych **heurystyką**).

5. Strategia poinformowana (z heurystyką)

W strategii **poinformowanego przeszukiwania** rozpatruje się funkcję oceny węzła (*koszt*, ale może też być dualnie maksymalizowany *zysk*) złożoną z 2 części: $f(n) = g(n) + h(n)$,

gdzie $g(n)$ oznacza koszt dotychczasowej ścieżki a $h(n)$ oznacza **przewidywany koszt** na drodze pozostałej z węzła *n* do celu.

W zależności od postaci funkcji oceny rozpatruje się dwie główne strategie:

1. strategia **zachłanna** („najbliższy celowi najpierw”), $f(n) = h(n)$,
2. **przeszukiwanie A*** , $f(n) = g(n) + h(n)$.

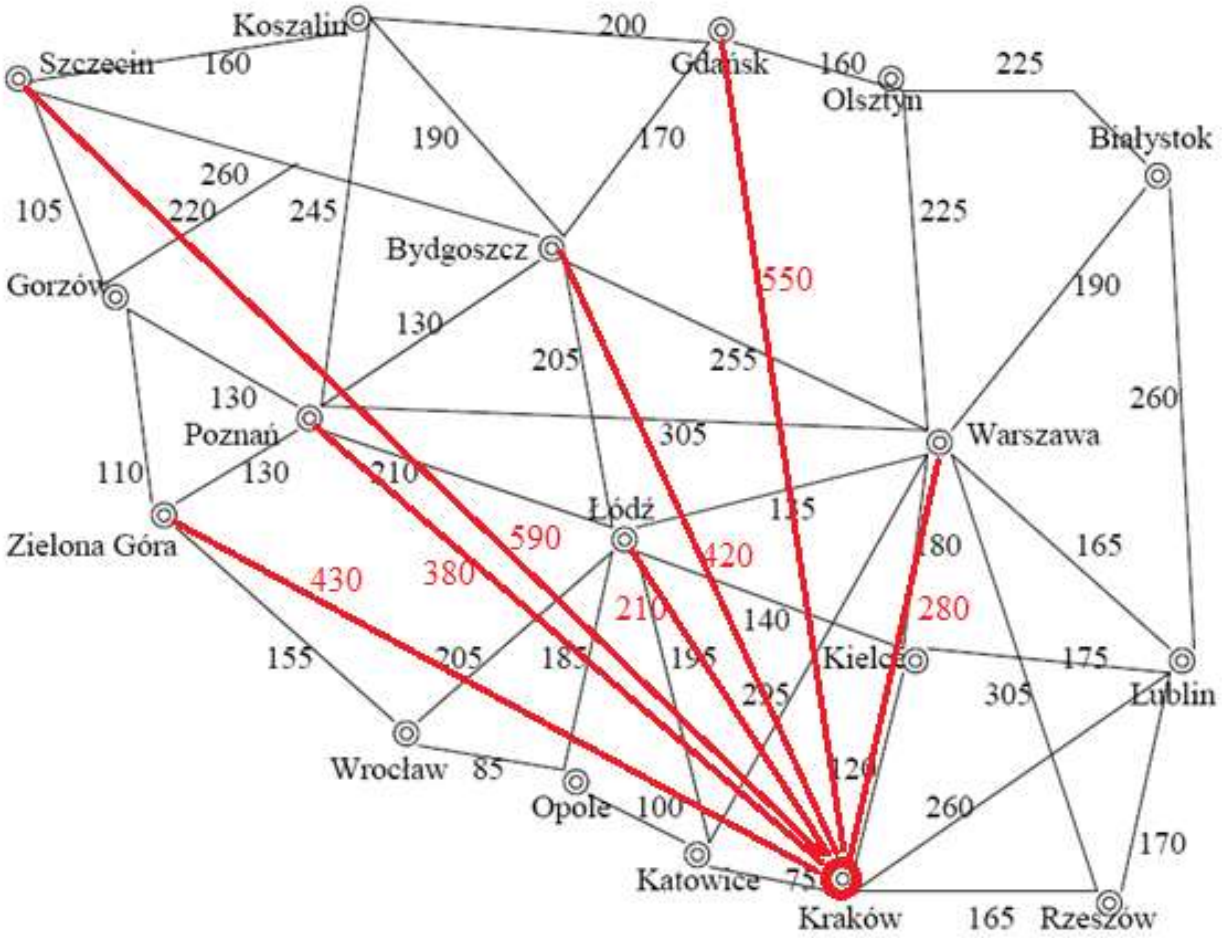
Obie strategie 2) i 3) są **„poinformowane”** (przesądza o tym składowa kosztu h(n)).

Oszacowanie kosztów resztkowych (heurystyka)

- **Szczególne przypadki** przeszukiwania poinformowanego (jako podkategorii przeszukiwania „***best first***”) polegają na wykorzystaniu **oszacowania** kosztów resztkowych (nazywanego **heurystyką**) dla każdego węzła w drzewie przeszukiwania (drzewie decyzyjnym).
- Obie strategie przeszukiwania poinformowanego („strategia zachłanna”, A*), poza wspomnianą różnicą w funkcji kosztu:
  - stosują **skraj globalny** (lista OPEN),
  - przeznaczone są dla **grafów** (lista CLOSED) unikając powielania równoważnych węzłów.
- Optymalność wyniku przeszukiwania (dla dopuszczalnej heurystyki) zapewnia jedynie strategia A*.

Przykład heurystyki: odległość w linii prostej od celu

Mapa rzeczywistych odległości cząstkowych:



Oszacowanie $h(n)$: odległość w prostej linii do celu (z n do Krakowa)

Białystok	440
Bydgoszcz	420
Gdańsk	550
Gorzów	500
Katowice	70
Kielce	110
Koszalin	580
Kraków	0
Lublin	230
Łódź	210
Opole	150
Poznań	380
Rzeszów	150
Olsztyn	460
Szczecin	590
Warszawa	280
Wrocław	240
Zielona Góra	430

Pytania

1. Jak definiujemy problem przeszukiwania?
2. Na czym polegają strategie ślepego przeszukiwania? Wymienić główne strategie i zilustrować je na przykładzie.
3. Przy pomocy jakich kryteriów porównujemy ze sobą strategie przeszukiwania?
4. Omówić problemy przeszukiwania drzewa a grafu.
5. Porównać ze sobą strategie ślepego przeszukiwania.
6. Wyjaśnić pojęcie przeszukiwania **„najpierw najlepszy”** („best first”).
7. Wyjaśnić pojęcie przeszukiwania **„poinformowanego”** („z heurystyką”).