

Algoritam 1: Pretraživanje po širini (engl. *breadth-first*)

1. Formirati listu čvorova koja inicijalno sadrži samo startni čvor.
2. Dok se lista čvorova ne isprazni ili se ne dođe do ciljnog čvora, proveriti da li je prvi element liste ciljni čvor
 - 2.1. Ako je prvi element ciljni čvor, ne raditi ništa.
 - 2.2. Ako prvi element nije ciljni čvor, ukloniti prvi element iz liste i dodati njegove sledbenike iz stabla pretrage (ako ih ima) *na kraj* liste.
3. Ako je pronađen ciljni čvor, pretraga je uspešno završena; u suprotnom pretraga je neuspešna.

Algoritam 2: Pretraživanje po dubini (engl. *depth-first*)

1. Formirati listu čvorova koja inicijalno sadrži samo startni čvor.
2. Dok se lista čvorova ne isprazni ili se ne dođe do ciljnog čvora, proveriti da li je prvi element liste ciljni čvor
 - 2.1. Ako je prvi element ciljni čvor, ne raditi ništa.
 - 2.2. Ako prvi element nije ciljni čvor, ukloniti prvi element iz liste i dodati njegove sledbenike iz stabla pretrage (ako ih ima) *na početak* liste.
3. Ako je pronađen ciljni čvor, pretraga je uspešno završena; u suprotnom pretraga je neuspešna.

Algoritam 3: Pretraživanje metodom planinarenja (engl. *hill-climbing*)

1. Formirati listu čvorova koja inicijalno sadrži samo startni čvor.
2. Dok se lista čvorova ne isprazni ili se ne dođe do ciljnog čvora, proveriti da li je prvi element liste ciljni čvor
 - 2.1. Ako je prvi element ciljni čvor, ne raditi ništa.
 - 2.2. Ako prvi element nije ciljni čvor, ukloniti prvi element iz liste, sortirati njegove sledbenike iz stabla pretrage (ako ih ima) po rastućim vrednostima heurističke funkcije. Zatim te sledbenike dodati *na početak* liste tako da prvi element liste bude sledbenik sa najmanjom vrednošću heurističke funkcije.
3. Ako je pronađen ciljni čvor, pretraga je uspešno završena; u suprotnom pretraga je neuspešna.

Algoritam 4: Pretraživanje metodom prvi najbolji (engl. *best-first*)

1. Formirati listu čvorova koja inicijalno sadrži samo startni čvor.
2. Dok se lista čvorova ne isprazni ili se ne dođe do ciljnog čvora, proveriti da li je prvi element liste ciljni čvor
 - 2.1. Ako je prvi element ciljni čvor, ne raditi ništa.
 - 2.2. Ako prvi element nije ciljni čvor, ukloniti prvi element iz liste i dodati njegove sledbenike iz stabla pretrage (ako ih ima) u listu. Celokupnu listu sortirati po rastućim vrednostima heurističkih funkcija čvorova.
3. Ako je pronađen ciljni čvor, pretraga je uspešno završena; u suprotnom pretraga je neuspešna.

Algoritam 5: Pretraživanje metodom grananja i ograničavanja (engl. *branch and bound*)

1. Formirati listu parcijalnih putanja. Inicijalno lista sadrži samo jednu putanju nulte dužine koja sadrži samo startni čvor.
2. Dok se lista čvorova ne isprazni ili se ne dođe do ciljnog čvora, proveriti da li je prvi element liste putanja koja dostiže ciljni čvor.
 - 2.1. Ako je prva putanja dostigla ciljni čvor, ne raditi ništa.
 - 2.2. Ako prva putanja nije dostigla ciljni čvor, uraditi sledeće:
 - 2.2.1. Ukloniti prvu putanju iz liste.
 - 2.2.2. Za svaki sledbenik poslednjeg čvora na uklonjenoj putanji formirati po jednu novu putanju produžujući sledbenikom uklonjenu putanju.
 - 2.2.3. Za svaku od novodobijenih putanja izračunati ukupnu (kumulativnu) cenu koštanja c kao zbir cena koštanja operatora na toj putanji.
 - 2.2.4. Dodati nove putanje u listu parcijalnih putanja.
 - 2.2.5. Sortirati listu putanja po rastućim vrednostima cena koštanja putanja.
3. Ako je pronađen ciljni čvor, pretraga je uspešno završena; u suprotnom pretraga je neuspešna.

Algoritam 6: Pretraživanje metodom A*

1. Formirati listu parcijalnih putanja. Inicijalno lista sadrži samo jednu putanju nulte dužine koja sadrži samo startni čvor.
2. Dok se lista čvorova ne isprazni ili se ne dođe do ciljnog čvora, proveriti da li je prvi element liste putanja koja dostiže ciljni čvor.
 - 2.1. Ako je prva putanja dostigla ciljni čvor, ne raditi ništa.

- 2.2. Ako prva putanja nije dostigla ciljni čvor, uraditi sledeće:
 - 2.2.1. Ukloniti prvu putanju iz liste.
 - 2.2.2. Za svaki sledbenik poslednjeg čvora na uklonjenoj putanji formirati po jednu novu putanju produžujući sledbenikom uklonjenu putanju.
 - 2.2.3. Za svaku od novodobijenih putanja izračunati ukupnu (kumulativnu) cenu koštanja c kao zbir cena koštanja operatora na toj putanji; za poslednji čvor na putanji izračunati heurističku funkciju h . Funkciju procene f za svaku od novih putanja izračunati kao zbir heurističke funkcije h i cene koštanja putanje c ($f = h + c$).
 - 2.2.4. Dodati nove putanje u listu parcijalnih putanja.
 - 2.2.5. Sortirati listu putanja po rastućim vrednostima funkcije procene f .
 - 2.2.6. Ako dve ili više putanja iz liste imaju isti poslednji čvor, ukloniti iz liste sve takve putanje osim jedne koja ima najmanju cenu koštanja (princip dinamičkog programiranja).
3. Ako je pronađen ciljni čvor, pretraga je uspešno završena; u suprotnom pretraga je neuspešna.

Algoritam 8: AO*

1. Graf pretrage G sastoji se inicijalno samo od startnog čvora n_0 . Funkcija procene f za startni čvor jednaka je njegovoj heurističkoj funkciji $f(n_0) = h(n_0)$.
2. Sve dok startni čvor ne bude obeležen kao REŠEN, raditi sledeće:
 - 2.1. Odrediti graf parcijalnog rešenja G' sledeći markirane konektore u grafu pretrage G od startnog čvora. Inicijano G' se sastoji samo od startnog čvora. Izabrati (na proizvoljan način) neki od nerazvijenih čvorova iz G' .
 - 2.2. Razviti izabrani čvor n . Ako se neki od čvorova naslednika čvora n ne nalazi u grafu, uneti ga u graf i vrednost njegove funkcije procene f postaviti na vrednost heurističke funkcije toga čvora. Ako uneti čvor predstavlja jednog od ciljnih čvorova, obeležiti ga kao REŠENOG.
 - 2.3. Ažurirati funkciju procene izabranog čvora n na sledeći način:
 - 2.3.1. Funkcija procene $f(n)$ čvora n jednaka je minimumu svih funkcija procene f_{K_i} po svakom od izlaznih konektora K_1, K_2 do K_p :

$$f(n) = \min(f_{K_1}, f_{K_2}, \dots, f_{K_p})$$

pri čemu se funkcija procene f_{K_i} izlaznog konektora K_i računa kao zbir cene c_{K_i} konektora K_i i (ranije izračunatih) vrednosti funkcija procene f_{n_j} svih čvorova n_1, n_2, \dots, n_m koji su konektorom K_i spojeni sa izabranim čvorom n :

$$f_{K_i} = c_{K_i} + f(n_1) + f(n_2) + \dots + f(n_m).$$

- 2.3.2. Obeležiti onaj konektor K_i među izlaznim konektorima izabranog čvora n koji ima minimalnu vrednost funkcije procene (prethodno ukloniti oznaku sa ranije obeleženog konektora, ako takav postoji).
- 2.3.3. Ako su svi čvorovi naslednici čvora n po izabranom konektoru K_i obeleženi kao REŠENI, obeležiti i čvor n kao REŠEN.
- 2.3.4. Ako je funkcija procene čvora n izračunata u koraku 2.3.1 različita (veća od) stare ili je čvor n u koraku 2.3.3. obeležen kao rešen, ponoviti korak 2.3. (ažurirati njihove funkcije procene i status rešenosti) za svaki od prethodnika čvora n po obeleženom konektoru u grafu G' .