d. Odstráň z OPEN všetky uzly, ktoré sú vyriešené, alebo ich predchodca je vyriešený.
6. Choď na (2).

Prehľadávanie AND/OR grafu do hĺbky – oproti predchádzajúcemu algoritmu stačí zmeniť krok (3) takto:

3' Ak je hĺbka uzla n menšia ako zvolená maximálna hĺbka, potom: Expanduj uzol n – vygeneruj všetkých jeho potomkov a pre každého potomka m, ak m reprezentuje množinu viac ako jedného subproblému, vygeneruj potomkov m (podľa charakteru daných subproblémov). Ku každému vygenerovanému uzlu pripoj smerník na jeho predchodcu (otca). Všetky nové uzly, ktoré zatiaľ nemajú potomkov, vlož na začiatok zoznamu OPEN.

Tento algoritmus nájde riešenie, ak existuje v rámci zvolenej maximálnej hĺbky; pre výpočet hĺbky na koniec kroku (3) možno ešte pridať:

Pre každý uzol x, ktorý sa pridáva do OPEN, vypočítaj jeho hĺbku ako hĺbku uzla n plus 1.

Ak je hĺbka počiatočného uzla 0, potom hĺbka ľubovoľného uzla x je dĺžka sekvencie operátorov, ktoré musíme aplikovať, aby sme sa dostali do uzla x.

6.2.3. Heuristické prehľadávanie stavového priestoru

Predpokladajme, že definície počiatočných stavov, operátorov a cieľových stavov sú dané. Otázkou je, ako prehľadávať takto definovaný stavový priestor efektívne. To si vyžaduje existenciu dodatočnej informácie o vlastnostiach danej problémovej oblasti (mimo tej, ktorá je obsiahnutá v definíciách stavov a operátorov). Informácia tohto druhu sa nazýva heuristická informácia a príslušné metódy heuristické metódy prehľadávania.

Väčšinu heuristických techník prehľadávania možno študovať ako variáciu metód slepého prehľadávania pre ten istý typ reprezentácie problému.

Heuristická informácia môže byť využitá pri prehľadávaní vo viacerých bodoch:

1. pri rozhodovaní, ktorý uzol expandovať ako ďalší
2. pri expandovaní uzla rozhodnúť – ktorého potomka alebo potomkov generovať – miesto "slepého" generovania všetkých možných potomkov
3. pri rozhodovaní, či dané uzly možno zanedbať alebo odseknúť zo stromu prehľadávania

Ďalej bude uvedený algoritmus, ktorý využíva heuristickú informáciu iba podľa 1. bodu, s tým, že daný uzol sa expanduje úplne alebo vôbec nie. Ako prvý sa vždy expanduje uzol, ktorý je "najsľubnejší". Prehľadávanie, realizované na tomto princípe, sa nazýva usporiadané (ordered search) alebo prehľadávanie typu "prvý-najlepší" (best-first search).

Využitie heuristickej informácie podľa 2. bodu vlastne spočíva v rozhodnutí, ktorý operátor aplikovať v danom uzle ako ďalší. Uzol, na ktorý boli aplikované niektoré, ale nie všetky operátory, sa nazýva parciálne rozvinutý (partially developed) alebo parciálne expandovaný (partially expanded). Dôležitým variantom tohto prístupu je myšlienka analýzy cieľov a prostriedkov - najprv sa vyberie operátor, ktorého aplikáciou sa pravdepodobne najviac priblížime k požadovanému cieľu, bez ohľadu na to, či je alebo nie je momentálne aplikovateľný. Úloha aplikovateľnosti daného operátora je potom druhotnou úlohou.

Tretí spôsob využitia heuristickej informácie - osekávanie (pruning) spočíva v rozhodnutí, ktoré uzly nebudú nikdy expandované. V niektorých prípadoch je možné jednoznačne rozhodnúť, že daný uzol nie je časťou riešenia, a preto ho možno odseknúť zo stromu prehľadávania. Odseknutie uzla môže byť žiadúce aj v iných prípadoch, hoci nie je zaručená ich nezávislosť od hľadaného riešenia (jedným dôvodom môže byť napr. úspora pamäti).

<u>Usporiadané prehľadávanie stavového priestoru</u> - voľba "najsľubnejšieho" uzla je väčšinou globálna, t.j. určí sa zo všetkých vygenerovaných, ale doteraz neexpandovaných uzlov. Bol by možný aj lokálny prístup - expanduje sa najsľubnejší uzol z posledne expandovaných uzlov.

Spôsobov, ako definovať "sľubnosť", prijateľnosť daného uzla (ako ho ohodnotiť) je niekoľko - napr. odhadnúť jeho vzdialenosť od cieľového uzla alebo predpokladať, že cesta riešenia zahŕňa daný uzol a ohodnotiť, resp. odhadnúť obtiažnosť celej cesty. Ohodnotenie uzla môže zvažovať iba niektoré preddefinované črty uzla, príp. môže stanoviť relevantné črty porovnaním daného uzla s cieľovým. Toto hodnotenie uzla sa robí pomocou <u>heuristickej funkcie</u> (heuristic function).

Ďalej uvedený algoritmus pre usporiadané prehľadávanie stavového priestoru bol navrhnutý Nilssonom (1971) [1]; hodnotiaca funkcia f* je navrhnutá tak, že čím je daný uzol sľubnejší, tým je hodnota f* menšia (pri expandovaní sa teda vyberá minimum). Stavový priestor sa predpokladá v tvare všeobecného grafu.

<u>Algoritmus:</u>

1. Vlož štartovací uzol <u>s</u> do zoznamu neexpandovaných uzlov OPEN. Vypočítaj hodnotu f*(s) a zviaž ju s uzlom <u>s</u>.
2. Ak je OPEN prázdny, algoritmus končí neúspechom.
3. Zo zoznamu OPEN vyber uzol <u>i</u>, ktorého hodnota f* je minimálna. Ak je takýchto uzlov viac, vyber ten, ktorý je cieľový (ak taký existuje), inak vyber náhodne.
4. Vlož uzol <u>i</u> do zoznamu expandovaných uzlov CLOSED.

5. Ak je i cieľový uzol, algoritmus končí úspechom, riešenie bol nájdené.
6. Expanduj uzol i, vytvor všetkých jeho potomkov a potom pr každého potomka j uzla i urob:
   a. Vypočítaj f*(j).
   b. Ak j nie je ani v zozname OPEN ani v CLOSED, pridaj ho do OPEN aj s jeho hodnotou f*. Pridaj smerník od j smerom na jeho otca - uzol i.
   c. Ak už bol j buď v OPEN alebo CLOSED, porovnaj nov vypočítanú hodnotu f* uzla j s už existujúcou. Ak j nová hodnota f* menšia, potom:
      i. nahraď ňou starú hodnotu f*
      ii. nastav smerník od j k i
      iii. ak už bol uzol j v CLOSED, presuň ho znovu do OPE
7. Choď na (2).

Krok (6c) je nevyhnutný iba pre všeobecné grafy (ak môže m uzol viacerých otcov), pre stromy môže byť tento krok vynechaný. Treba si všimnúť, že hoci je priestor prehľadávania všeobecn graf, explicitný subgraf, ktorý sa vytvára pri prehľadávaní, j stále strom (stále sa pamätá iba jeden otec).

Prehľadávanie do šírky, do hĺbky, s uniformnou cenou s všetko špeciálne prípady tohto algoritmu: pre prehľadávanie d šírky je f*(i) rovné hĺbke uzla i; pre prehľadávanie s uniform cenou je f*(i) rovné cene cesty z počiatočného uzla do uzla j prehľadávanie do hĺbky (bez obmedzenia hĺbky) získame, ak polo žíme f*(i) rovné negatívnej hodnote hĺbky uzla i.

Cieľom usporiadaného prehľadávania je znížiť počet expan dovaných uzlov (v porovnaní so slepým prehľadávaním), efektívno priamo závisí na voľbe f*. Ak je voľba f* nesprávna, usporiad ným prehľadávaním nemusíme nájsť optimálne riešenie, resp. vôbe žiadne riešenie. Ak neexistuje exaktný spôsob hodnotenia uzlov voľba f* zahŕňa porovnanie nárokov na čas a pamäť a garanciu pr nájdenie optimálneho, príp. vôbec nejakého riešenia.

Typy problémov a voľba hodnotiacej funkcie f* - závisia o daného problému, možno však rozlíšiť niekoľko typov problémov:
1. Predpokladá sa, že stavový priestor obsahuje viacero cie riešenia s rôznymi cenami. Cieľom je nájsť optimálnu cestu minimálnou cenou). Tento typ úloh je dobre rozpracovaný (vi algoritmus A*).
2. Problém je podobný, ako v predchádzajúcom prípade, avšak j zložitejší, a preto by pri prehľadávaní došlo k prekročeni požiadaviek na čas a pamäť. Kľúčové otázky sú:
   a. Ako nájsť dobré (ale nie optimálne) riešenie pri vyn ložení "rozumného množstva úsilia" na prehľadávanie.
   b. Ako ohraničiť toto množstvo úsilia aj mieru, do ktor je získané riešenie horšie ako optimálne.

3. Príklad tretieho typu úloh môžeme zobrať z oblasti dokazovania teorém, keď sa uspokojíme s najľahšie nájdeným dôkazom, aj keď nie je veľmi elegantný.

Príkladom úloh druhého typu môže byť úloha obchodného cestujúceho, aj keď sa niekedy prísne nerozlišuje medzi 1. a 2. typom úloh ("hra 8" môže byť zaradená medzi obidva tieto typy).

6.2.4. *Algoritmus A* - optimálne hľadanie optimálneho riešenia*

Algoritmus A*, ktorý vypracovali Hart, Nilsson, Raphael (1968) [1], je určený na nájdenie cesty minimálnej ceny z počiatočného do cieľového uzla v stavovom priestore. Tento problém zahŕňa úlohu nájsť cestu v grafe medzi dvoma uzlami s minimálnym počtom hrán (t.j. cena hrany je 1).

Algoritmus A* je založený na usporiadanom prehľadávaní. Hodnotiaca funkcia f* pozostáva z dvoch zložiek - hodnota každého uzla <u>n</u> je daná súčtom ceny cesty z počiatočného uzla do uzla <u>n</u> a z ceny cesty z <u>n</u> do cieľového uzla:

$$f^*(n) = g^*(n) + h^*(n) \quad , \tag{6.1}$$

kde $g^*(n)$ udáva odhad minimálnej ceny cesty z počiatočného uzla do uzla <u>n</u> a $h^*(n)$ odhad minimálnej ceny cesty z uzla <u>n</u> do cieľového uzla. Skutočné hodnoty, ktoré f*, g*, h* iba odhadujú, sú označené f, g, h. Predpokladá sa, že všetky ceny hrán sú kladné.

Funkcie g*, ktorej hodnota sa využíva pri expandovaní uzla, vypočítava skutočnú hodnotu cesty zo štartovacieho uzla <u>s</u> do uzla <u>n</u> po najlacnejšej ceste, doteraz nájdenej algoritmom. Ak je stavový priestor popísaný stromom, potom g* dáva perfektný odhad, pretože z <u>s</u> do <u>n</u> existuje jediná cesta. Ak sa jedná o všeobecný graf, odhad g* môže byť nadhodnotený - hodnotu g* možno zredukovať, ak sa nájde kratšia cesta do <u>n</u>. Avšak aj v tomto prípade, za istých podmienok, môže $g^*(n)$ dávať perfektný odhad v čase, keď sa <u>n</u> expanduje.

Funkcia h* je nositeľom heuristickej informácie, spôsob jej definície závisí od danej problémovej oblasti. Pre zabezpečenie určitých vlastností algoritmu A* sa však vyžaduje, aby to bola nezáporná funkcia a aby neprehodnocovala cenu cesty z <u>n</u> do cieľového uzla, t.j. odhad tejto ceny nesmie byť nikdy väčší ako skutočná cena cesty: $h^*(n) \leq h(n)$. Táto podmienka sa nazýva <u>podmienka prípustnosti</u> (admissibility condition).

<u>Prípustnosť a optimálnosť algoritmu A*</u> - možno dokázať, že ak h* spĺňa podmienku prípustnosti a naviac ceny všetkých hrán sú pozitívne ohraničené zdola kladným číslom, potom A* nájde optimálnu cestu, za predpokladu, že riešenie existuje. Táto vlast-

nosť sa nazýva vlastnosť prípustnosti (property of admissibility).

Hoci si podmienka prípustnosti vyžaduje, aby bola h^* dolnou hranicou h, čím presnejšie h^* aproximuje h, tým je algoritmus efektívnejší. Ak by bolo h^* rovné h, optimálne riešenie by bolo nájdené bez toho, aby bol expandovaný uzol, ktorý neleží na tejto optimálnej ceste. Ak $h^* \equiv 0$, algoritmus A* degeneruje na slepé prehľadávanie s uniformnou cenou. Dva algoritmy A1, A2 možno porovnať s ohľadom na voľbu funkcie h^*; nech algoritmus A1 zodpovedá $h1^*$ a algoritmus A2 nech zodpovedá $h2^*$. O algoritme A1 tvrdíme, že je informovanejší ako A2, ak pre ohodnotenie uzla n platí

$$h1^*(n) > h2^*(n) \qquad (6.2)$$

Na tomto princípe možno definovať optimálnosť riešenia: ak A a A* sú prípustné algoritmy také, že A* je informovanejší ako A, potom A* nikdy neexpanduje uzol, ktorý neexpanduje aj A.

Optimálnosť a heuristická sila (optimality and heuristic power) – optimálnosť prehľadávania algoritmom A* je daná iba počtom expandovaných uzlov pri hľadaní optimálneho riešenia. Ale dôležité môžu byť aj iné charakteristiky. Po prvé, zložitosť výpočtu h^* takisto ovplyvňuje celkovú výpočtovú zložitosť. Po druhé, niekedy je menej dôležité nájsť riešenie s absolútne minimálnou cenou, ako nájsť riešenie rozumnej ceny prehľadávaním za rozumný čas – v takom prípade možno použiť h^*, ktoré ohodnocuje uzly presnejšie, ale v určitých prípadoch môže nadhodnotiť vzdialenosť od cieľa (porušenie podmienky prípustnosti). Voľba h^* a výsledná heuristická sila závisí od kompromisu medzi týmito dvoma požiadavkami.

Dôležitý je aj pomer počtu expanzií uzla k počtu rôznych uzlov expandovaných algoritmom A*. Tieto dva počty budú rovnaké, ak zakaždým, keď je uzol n expandovaný (zaradený do zoznamu CLOSED), už bola nájdená optimálna cesta do n. Táto podmienka je splnená vždy, keď sa jedná o strom, vtedy nutne platí $g^*(n)=g(n)$. Platí to aj pre všeobecné grafy za predpokladu, že platí podmienka – predpoklad konzistencie (consistency assumption): pre všetky uzly m a n je odhadnutá vzdialenosť (cena cesty) $h^*(m)$ z m do cieľa vždy menšia alebo rovná skutočnej vzdialenosti z m do n plus odhad zvyšnej vzdialenosti z n do cieľa (t.j. vždy platí $h^*(m) \le c(m,n)+h^*(n)$). Martelli (1977) [1] dokázal, že ak h^* nespĺňa túto podmienku vo všeobecnom grafe, algoritmus A* nie je optimálny s ohľadom na počet expanzií a navrhol pre tento prípad efektívnejší algoritmus.

Na záver niekoľko otvorených otázok, týkajúcich sa algoritmu A*:

1. Niekedy je dôležitejšie minimalizovať prehľadávanie, ak... samotné riešenie. Je v tomto prípade $f^* = g^* + h^*$ vhodnou voľbou?

2. Aj keď cena riešenia je dôležitá, kombinatorická zložitosť problému môže byť taká veľká, že prípustný A* algoritmus nemôže dobehnúť do konca. Možno získať na rýchlosti za cenu určitého zhoršenia kvality riešenia?
3. Niekedy je ťažké nájsť dobrú heuristickú funkciu, spĺňajúcu podmienku prípustnosti; slabá, ale prípustná heuristická funkcia h* degeneruje A* na slepé prehľadávanie. Ako je prehľadávanie ovplyvnené voľbou neprípustnej heuristickej funkcie?

6.2.5. *Heuristické prehľadávanie AND/OR grafov*

Hlavný rozdiel oproti prehľadávaniu stavového priestoru spočíva v existencii AND uzlov, ktoré sťažujú prehľadávanie. Každý uzol AND/OR grafu reprezentuje cieľ, ktorý treba dosiahnuť. Budeme predpokladať spätné uvažovanie od počiatočného uzla (koreňa) k množine podcieľov. AND/OR grafy sú v tomto zmysle použité na reprezentáciu redukcie problému (rozdiel oproti stavovému priestoru spočíva aj v tom, že operátory stavového priestoru majú jeden vstup a jeden výstup a operátory redukcie problému môžu mať jeden vstup a niekoľko výstupov).

Môžeme uvažovať aj prehľadávanie AND/OR grafov v priamom smere od triviálnych problémov k problému, ktorý chceme vyriešiť (cieľ) - typický problém pri dokazovaní teorém.

<u>Definícia optimálneho riešenia</u> - cena stromu riešenia môže byť definovaná dvoma spôsobmi:

1. Celková cena stromu riešenia je daná ako suma cien (váh) všetkých hrán stromu.
2. Maximálna cena stromu riešenia je suma cien pozdĺž najdrahšej cesty z koreňa do terminálneho uzla.

Ak je cena každej hrany 1, celková cena stromu sa rovná počtu hrán stromu a maximálna cena hĺbke najhlbšieho uzla.

Nech $c(n,m)$ je cena hrany z uzla <u>n</u> do uzla <u>m</u>. Definujme funkciu $h(n)$ takto:

1. Ak je <u>n</u> terminálny uzol (triviálny problém), potom $h(n)=0$.
2. Ak má <u>n</u> OR potomkov, potom $h(n)$ je minimum cez všetkých potomkov <u>m</u> z $c(n,m)+h(m)$.
3. Ak má <u>n</u> AND potomkov a využíva sa celková cena, potom $h(n)$ je suma cez všetkých potomkov <u>m</u> z $c(n,m)+h(m)$.
4. Ak má <u>n</u> AND potomkov a využíva sa maximálna cena, potom $h(n)$ je maximum cez všetkých potomkov <u>m</u> z $c(n,m)+h(m)$.
5. Ak je <u>n</u> neterminálny uzol (netriviálny problém) a nemá potomkov, potom $h(n)$ je nekonečno.

Je zrejmé, že $h(n)$ je konečné, iba ak je problém reprezen-