Zaawansowane programowanie

wykład 3: inne heurystyki

prof. dr hab. inż. **Marta Kasprzak**
Instytut Informatyki, Politechnika Poznańska

Heurystyki

- *Heurystyką* nazywamy algorytm (metodę) zwracający rozwiązanie przybliżone. Zazwyczaj oczekuje się, że algorytm taki ma złożoność wielomianową i działa szybko w praktyce
- Metoda heurystyczna dla problemów optymalizacyjnych nie gwarantuje znalezienia optymalnego rozwiązania. Powinna dążyć do wygenerowania rozwiązania dopuszczalnego o wartości funkcji celu jak najbliższej optymalnej
- W przypadku dowolnych problemów przeszukiwania metoda heurystyczna może nie zwrócić żadnego rozwiązania, jeśli nie znajdzie dopuszczalnego. Może także (w zależności od potrzeby) naruszyć ograniczenie na dopuszczalność rozwiązania. Powinna wtedy dążyć do wygenerowania rozwiązania jak najbardziej podobnego do rozwiązania dopuszczalnego (np. jak najdłuższa prosta ścieżka w grafie zamiast ścieżki Hamiltona)

Heurystyki

- Proste heurystyki, jak np. algorytm zachłanny, stosuje się powszechnie do generowania rozwiązań początkowych, poprawianych następnie przez bardziej złożone algorytmy
- Przykładem algorytmów zyskujących na dobrym rozwiązaniu początkowym są metaheurystyka *tabu search* i algorytm dokładny podziału i ograniczeń. W ich przypadku rozpoczynanie poszukiwania od rozwiązania losowego znacznie wydłuża czas obliczeń (dla algorytmu dokładnego) lub zmniejsza jakość osiągniętych rozwiązań (dla metaheurystyki)
- Bardziej złożone heurystyki konstruktywne funkcjonują samodzielnie, osiągając dobre rozwiązania. Są to najczęściej bardzo specjalizowane (dostosowane do charakteru rozwiązywanego problemu) metody

Heurystyki

Heurystyki można podzielić na:

- *Budujące*/*konstruktywne* (ang. *constructive heuristics*) — konstruują rozwiązanie krok po kroku. Pierwsze i jedyne rozwiązanie otrzymywane jest wraz z zakończeniem procedury
  - Heurystyka zachłanna
  - Przeszukiwanie wiązkowe
  - Metoda dekompozycji problemu
- *Polepszające* (ang. *improvement/perturbative heuristics*) — rozpoczynają działanie od rozwiązania dopuszczalnego i w kolejnych krokach modyfikują je
  - Przeszukiwanie lokalne
  - Metaheurystyki

Heurystyka zachłanna

- *Heurystyka zachłanna* (ang. *greedy heuristic*) polega na sukcesywnej budowie rozwiązania poprzez podejmowanie w każdym kroku optymalnej w danym momencie decyzji
- W problemie optymalizacyjnym decyzja najczęściej polega na wyborze takiego elementu rozwiązania, który optymalizuje wartość funkcji celu dla bieżącego rozwiązania
- Rozwiązanie konstruowane jest „od zera” i tylko jednokrotnie — żadna z podjętych decyzji nie jest zmieniana
- Pomimo dużej prostoty podejście to jest dość skuteczne, stąd jest często stosowane do generowania rozwiązań początkowych w złożonych heurystykach

Heurystyka zachłanna

Przykład heurystyki zachłannej dla problemu komiwojażera

- W pierwszym kroku do pustego rozwiązania dodawane jest dowolne miasto. W każdym kolejnym kroku dodawane jest to z nieodwiedzonych miast, którego odległość od miasta bieżącego jest najmniejsza. Na końcu trasa jest domykana

	A	B	C	D	E
A	0	18	32	27	11
B	18	0	49	34	15
C	32	49	0	37	35
D	27	34	37	0	23
E	11	15	35	23	0

	wartość funkcji celu
A	0
A E	11
A E B	26
A E B D C	129

Heurystyka zachłanna

Przykład heurystyki zachłannej dla problemu komiwojażera

- Rozpoczynając konstrukcję rozwiązania od innego miasta, możemy otrzymać inne rozwiązanie

	A	B	C	D	E
A	0	18	32	27	11
B	18	0	49	34	15
C	32	49	0	37	35
D	27	34	37	0	23
E	11	15	35	23	0

	wartość funkcji celu
B E A D C	139
C A E B D	129
D E A B C	138
E A B D C	135

- Optymalnym rozwiązaniem jest: A B E D C 125

7

Heurystyka zachłanna

- Decyzję o wyborze elementu można oprzeć na nieco dalszym spojrzeniu w przyszłość, czyniąc heurystykę nieco bardziej złożoną czasowo. Przykładowo, w problemie komiwojażera można dołączać do bieżącego rozwiązania takie miasto, które wraz z innym nieodwiedzonym miastem tworzy najoptymalniejszą parę. Nie oznacza to dołączenia od razu pary miast

	A	B	C	D	E
A	0	18	32	27	11
B	18	0	49	34	15
C	32	49	0	37	35
D	27	34	37	0	23
E	11	15	35	23	0

	wartość funkcji celu
C	0
C A E	32
C A E B	43
C A E B D	129

8

Heurystyka zachłanna

- Decyzja podejmowana w każdym kroku skutkuje dodaniem kolejnego elementu do bieżącego rozwiązania. Jednak nawet w ramach jednego problemu element taki można zdefiniować na różne sposoby. Przykładem może być — alternatywny do wyboru miasta — wybór w problemie komiwojażera najkrótszego możliwego odcinka pomiędzy miastami, który nie powoduje przedwczesnego zamknięcia ani rozwidlenia trasy

	A	B	C	D	E
A	0	18	32	27	11
B	18	0	49	34	15
C	32	49	0	37	35
D	27	34	37	0	23
E	11	15	35	23	0

A-E	11
A-E-B	26
D-A-E-B	53
D-A-E-B-C-D	139

9

Heurystyka przeszukiwania wiązkowego

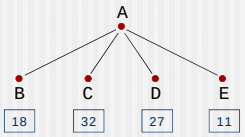
- Heurystyka *przeszukiwania wiązkowego* (ang. *beam search*) polega na ograniczeniu liczby odwiedzanych węzłów drzewa przeszukiwanego wszerz (ang. *breadth-first search*)
- Zazwyczaj pełen przegląd drzewa konstruowanych rozwiązań zajmuje wykładniczą względem rozmiaru instancji liczbę kroków. Heurystyka korzysta z systematyczności takiego przeglądu, ograniczając przy tym wielomianowo liczbę kroków
- Na każdym poziomie drzewa generowane są wszystkie następniki węzłów z tego poziomu i sortowane po malejącej wartości osiąganych w nich rozwiązań cząstkowych. Przeszukiwanie jest kontynuowane jedynie dla pewnej stałej liczby najlepszych węzłów z początku tej listy. Liczba ta nazywana jest szerokością wiązki (ang. *beam width*)

10

Heurystyka przeszukiwania wiązkowego

Przykład dla problemu komiwojażera z szerokością wiązki równą 3

	A	B	C	D	E
A	0	18	32	27	11
B	18	0	49	34	15
C	32	49	0	37	35
D	27	34	37	0	23
E	11	15	35	23	0

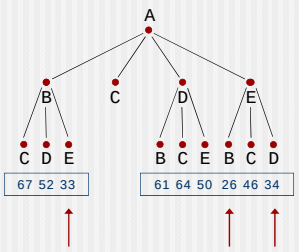


Węzły posortowane po malejącej wartości rozwiązań cząstkowych (tutaj po rosnącym koszcie trasy): **E**, **B**, **D**, C

11

Heurystyka przeszukiwania wiązkowego

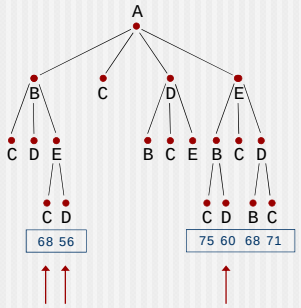
	A	B	C	D	E
A	0	18	32	27	11
B	18	0	49	34	15
C	32	49	0	37	35
D	27	34	37	0	23
E	11	15	35	23	0



12

Heurystyka przeszukiwania wiązkowego

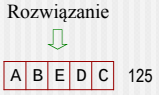
	A	B	C	D	E
A	0	18	32	27	11
B	18	0	49	34	15
C	32	49	0	37	35
D	27	34	37	0	23
E	11	15	35	23	0

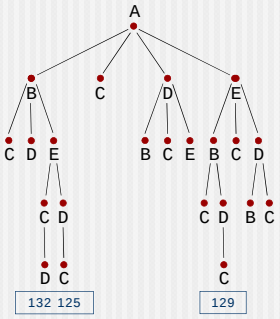


Heurystyka przeszukiwania wiązkowego

	A	B	C	D	E
A	0	18	32	27	11
B	18	0	49	34	15
C	32	49	0	37	35
D	27	34	37	0	23
E	11	15	35	23	0

Rozwiązanie





Heurystyka dekompozycji problemu

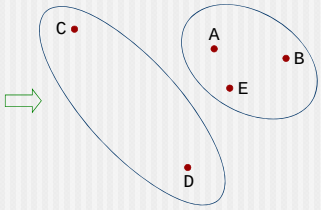
- Heurystyka *dekompozycji problemu* (ang. *problem decomposition*) polega na podziale złożonego problemu na mniejsze. Podproblemy rozwiązywane są optymalnie i następnie heurystycznie łączone w rozwiązanie dopuszczalne problemu nadrzędnego
- Wyróżnione są trzy ogólne podejścia do dekompozycji problemu:
  - Podproblemy rozwiązywane są niezależnie
  - Podproblemy rozwiązywane są strumieniowo — po kolei, z przekazywaniem rozwiązania podproblemu poprzedniego na wejście następnego
  - Podproblemy rozwiązywane są iteracyjnie — dopuszcza się powrót do rozwiązanego już podproblemu

Heurystyka dekompozycji problemu

Przykład dla problemu komiwojażera

- Dekompozycja może polegać na podziale całego odwiedzanego obszaru na mniejsze

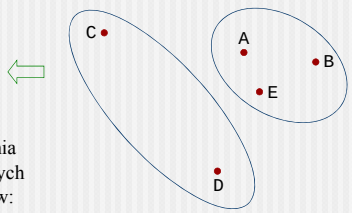
	A	B	C	D	E
A	0	18	32	27	11
B	18	0	49	34	15
C	32	49	0	37	35
D	27	34	37	0	23
E	11	15	35	23	0



Heurystyka dekompozycji problemu

- Podproblemy rozwiązywane są niezależnie

Rozwiązania cząstkowe:
A-B-E-A C-D-C



Rozwiązanie całościowe przy założeniu przerywania cykli w okolicy najbliższych miast sąsiednich obszarów:

A-B-E-A + C-D-C = A-B-C-D-E-A 138
A-B-E-A + C-D-C = A-B-E-D-C-A 125

Heurystyka dekompozycji problemu

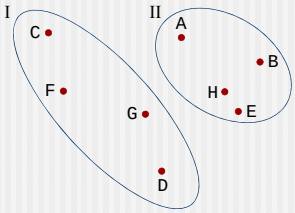
- Podproblemy rozwiązywane są strumieniowo

Wariant A: C-F-G-D-C

C-F-G-D-C + A-B-H-E-A =
= A-B-H-E-D-G-F-C-A

Wariant B: C-G-D-F-C

C-G-D-F-C + A-B-E-H-A =
= A-B-E-H-G-D-F-C-A



Rozwiązanie podproblemu zależy od postaci rozwiązania uzyskanego w poprzednim kroku

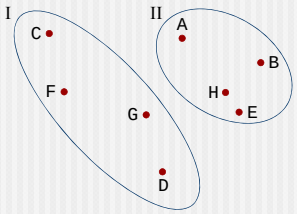
Heurystyka dekompozycji problemu

- Podproblemy rozwiązywane są iteracyjnie

Etap II:

C-G-D-F-C + A-B-E-H-A =

= A-B-E-H-G-D-F-C-A



Korekta etapu I:

C-F-G-D-C + A-B-E-H-A =

= A-B-E-H-D-G-F-C-A

Powrót do etapu II i korekta:

C-F-G-D-C + A-B-H-E-A = A-B-H-E-D-G-F-C-A

19

Heurystyka lokalnego przeszukiwania

- Heurystyka *lokalnego przeszukiwania* (ang. *local search*) polega na iteracyjnym poprawianiu bieżącego rozwiązania metodą drobnych kroków. W każdej iteracji wybierany jest sąsiad bieżącego rozwiązania o nie gorszej wartości funkcji celu
- Heurystyka startuje z rozwiązaniem początkowym wygenerowanym np. losowo. Zatrzymuje się, gdy wszyscy sąsiedzi są gorsi od bieżącego rozwiązania
- Brak etapu dywersyfikacji w tej metodzie powoduje osiągnięcie jedynie optimum lokalnego (będącego niekiedy także optimum globalnym)
- Aby zagwarantować wielomianową złożoność heurystyki, niekiedy należy ustalić dodatkowy warunek stopu

20

Heurystyka lokalnego przeszukiwania

- Sposób reprezentacji rozwiązania oraz definicja ruchów podlegają takim samym zasadom, jak w metaheurystyce tabu (wykorzystującej przeszukiwanie lokalne)
- Podobnie jest z praktyką ograniczania sąsiedztwa bieżącego rozwiązania, jeśli jest zbyt liczne. Inaczej jednak niż w metodzie tabu, tutaj znaczne ograniczenie sąsiedztwa może spowodować przedwczesne zatrzymanie się algorytmu
- Heurystykę lokalnego przeszukiwania można urozmaicić, stosując różne strategie wyboru sąsiada

21

Heurystyka lokalnego przeszukiwania

Strategie wyboru sąsiada bieżącego rozwiązania

- *Wybór najlepszego sąsiada* wymaga przejrzenia całego dostępnego sąsiedztwa, stąd często jest stosowany wraz z ograniczaniem sąsiedztwa
- *Wybór pierwszego sąsiada* poprawiającego wartość funkcji celu znacznie skraca czas przeszukiwania sąsiedztwa. Kolejność stosowania ruchów jest deterministyczna
- *Wybór losowy* polega na losowym wyborze ruchu modyfikującego bieżące rozwiązanie. Wybranym sąsiadem może być tylko ten, który nie ma gorszej wartości funkcji celu

22

Heurystyka lokalnego przeszukiwania

Przykład heurystyki lokalnego przeszukiwania dla problemu komiwojażera

- Rozwiązanie początkowe:

B	E	D	A	C

- Ruch zdefiniowany jako zamiana miejscami dwóch sąsiednich elementów, wybierany przypadkowy ruch przynoszący poprawę

	A	B	C	D	E
A	0	18	32	27	11
B	18	0	49	34	15
C	32	49	0	37	35
D	27	34	37	0	23
E	11	15	35	23	0

B	E	D	A	C	146	D-A
B	E	A	D	C	139	B-E
E	B	A	D	C	132	D-C
E	B	A	C	D	125	

23

Heurystyka lokalnego przeszukiwania

Warianty lokalnego przeszukiwania uzupełnione o dywersyfikację, mające znamiona metaheurystyki:

- *Multistart local search* — lokalne przeszukiwanie uruchamiane wielokrotnie dla kolejnych losowo wygenerowanych rozwiązań początkowych
- *Guided local search* — lokalne przeszukiwanie uruchamiane wielokrotnie ze zmodyfikowaną funkcją celu. Funkcję uzupełnia się o kary ułatwiające wyjście z obszaru lokalnego optimum. W trakcie obliczeń zliczane są „cechy" (elementy) rozwiązania, które występują w rozwiązaniach lokalnie optymalnych. Im więcej razy cecha wystąpi w takich rozwiązaniach, tym większa kara odpowiada jej w funkcji celu, która obowiązuje po restarcie

24

Heurystyka lokalnego przeszukiwania

- *Iterated local search* — lokalne przeszukiwanie uruchamiane wielokrotnie dla kolejnych rozwiązań początkowych będących modyfikacją najlepszego dotąd rozwiązania
  - Po osiągnięciu lokalnego optimum modyfikuje się najlepsze dotąd rozwiązanie poprzez zastosowanie serii ruchów, losowych lub deterministycznych. Otrzymane nowe rozwiązanie jest początkowym w kolejnym etapie
  - Jeśli eksploracja nowego obszaru przestrzeni rozwiązań zakończy się osiągnięciem lepszego wyniku, stanie się on podstawą nowego rozwiązania początkowego. W przeciwnym razie podstawą będzie ponownie wcześniejsze rozwiązanie
  - Modyfikacja nie powinna być zbyt mała, gdyż nie umożliwi wyjścia z obszaru lokalnego optimum, ani zbyt duża, gdyż uczyni przeszukiwanie zbyt losowym

25

Hiperheurystyka

- *Hiperheurystyka* (ang. *hyper-heuristic*) jest algorytmem nadrzędnym zarządzającym heurystykami z niższego poziomu. Hiperheurystyka tak steruje ich wykonaniem, aby uzyskać jak najlepsze rozwiązanie wynikowe
- Heurystyki niższego poziomu dedykowane są do rozwiązania konkretnego problemu optymalizacyjnego. Są to zazwyczaj bardzo proste metody, które dopiero w połączeniu w podejście hiperheurystyczne cechują się wysoką skutecznością
- Dobrze skonstruowana hiperheurystyka osiąga dobre wyniki dla szerokiego spektrum problemów optymalizacyjnych. Algorytmy niższego poziomu zamieniane są na inne, rozwiązujące nowy problem

26

Hiperheurystyka

- Hiperheurystyka nie ma wglądu w postać rozwiązania generowanego na niższym poziomie. Otrzymuje jedynie informację zwrotną, jaka wartość funkcji celu została osiągnięta przez kolejny algorytm. Czyni ją to w dużym stopniu niezależną od rodzaju rozwiązywanego problemu
- Heurystyki niższego poziomu mogą przybierać skrajnie prostą postać, np. zamiana miejscami pary elementów rozwiązania. W takim przypadku kolejne wywołania takich metod przypominają kolejne kroki przeszukiwania lokalnego. Heurystyką niższego poziomu może być także metaheurystyka
- Hiperheurystyka, jako metoda z elementami dywersyfikacji, może być postrzegana jako metaheurystyka, która jednak nie operuje na przestrzeni rozwiązań problemu

27

Hiperheurystyka

- W zamierzeniu podejście hiperheurystyczne ma wiązać się ze stosunkowo małym kosztem — niewielkim nakładem pracy, dającym jednak satysfakcjonujące rezultaty. Większy nakład pracy, profitujący jednak w przyszłości, spodziewany jest dla hiperheurystyki szerokiego zastosowania
- Często wykorzystuje się istniejące algorytmy specjalizowane dla danego problemu bądź ich części. W razie potrzeby implementacji nowych algorytmów są nimi raczej proste podejścia
- Dodatkowy nakład pracy jest zazwyczaj niezbędny w celu zapewnienia spójnego standardu komunikacji (jednolita postać wyniku zwracanego przez metody niższego poziomu)

28

Hiperheurystyka

- Sterowanie algorytmami niższego poziomu może odbywać się na podstawie predeterminowanego schematu uruchomień. Jednak efektywny schemat uzyskuje się raczej przez zaimplementowanie procesu uczenia się w hiperheurystyce
- Uczenie się polega na wykorzystaniu pamięci o zdarzeniach od początku uruchomienia hiperheurystyki do zbudowania skuteczniejszego schematu wyboru heurystyk i kolejności ich uruchamiania, liczby uruchomień czy określenia ich cykli
- Pamięć w hiperheurystyce może gromadzić np. informacje o dotychczasowej efektywności heurystyk, efektywności dla par heurystyk wywoływanych bezpośrednio po sobie, odstępach, w jakich ta sama heurystyka daje najlepsze rezultaty, itp.

29

Hiperheurystyka

- Wybór metody uruchamianej w danym kroku jest oparty także na stanie rozwoju bieżącego rozwiązania, np. na liczbie dotychczasowych iteracji, przyroście wartości funkcji celu
- Wykorzystanie rzadkich zdarzeń może być elementem strategii dywersyfikacji

30