Laboratorium Algorytmów i Struktur Danych			
Kierunek:Informatyka	Specjalność: -	Rok: I	Grupa: I2
Programowanie dynamiczne *(30.05.2017r.)*			Nr 5
Kinga Karasiewicz (*132247*)			
Uwagi:			Ocena:

1. Cel ćwiczenia

Celem laboratorium było zbadanie właściwości i skuteczności obliczania rozwiązania dla problemu plecakowego. Do rozwiązania wykorzystałam programowanie dynamiczne (PD), metodę „brutal force" (BF1), a także ten sam algorytm z odcięciami (BF2) oraz metody heurystyczne: losową (GH1), minimum względem ciężaru (GH2), maksimum względem wartości (GH3) i maksymalną zależność pomiędzy wartością i masą (GH4). Ćwiczenie polega na zmierzeniu czasu dla tych metod w zależności od liczby paczek (N), ciężar $s(a_i)$ jest losowany z przedziału <10,100>,a wartość $w(a_i)$ z przedziału <100,1000>, zaś pojemność plecaka zależy od sumy ciężaru elementów pomnożonego przez współczynnik d.

2. Podstawy teoretyczne

Problem plecakowy - jest przykładem zagadnienia optymalizacyjnego, które może być rozwiązywanie za pomocą wielu technik algorytmicznych. Dla tego problemu nie istnieje (przynajmniej nie został jeszcze odkryty) algorytm rozwiązujący dyskretny problem plecakowy w czasie wielomianowy. Nazwa zagadnienia pochodzi od maksymalizacyjnego problemu wyboru przedmiotów, tak by ich sumaryczna wartość była jak największa i jednocześnie mieściły się w plecaku. Przy podanym zbiorze elementów o podanej wadze i wartości, należy wybrać taki podzbiór by suma wartości była możliwie jak największa, a suma wag była nie większa od danej pojemności plecaka.

Programowanie dynamiczne – programowanie dynamiczne polega na stworzeniu tabeli pomocniczej, w której zapisywane są wyniki obliczeń, które są wykorzystywane w kolejnych krokach algorytmu, co eliminuje potrzebę wielokrotnego wykonywania tych samych obliczeń. Programowanie dynamiczne jest zazwyczaj stosowane w rozwiązywaniu problemów optymalizacyjnych, prowadzi to często do wyznaczenia kilku równoznacznych, optymalnych rozwiązań.

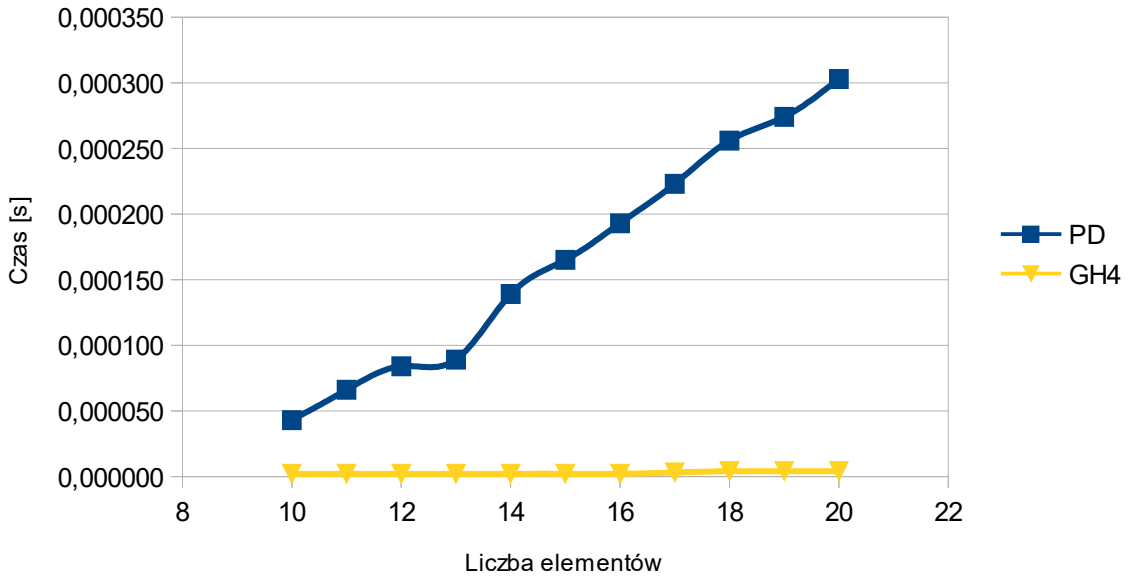
Metoda heurystyczna – jest to metoda znajdowania rozwiązań, dla której nie ma gwarancji znalezienia rozwiązania optymalnego, a często nawet prawidłowego. Rozwiązań tych używa się na przykład wtedy, gdy pełny algorytm jest z przyczyn technicznych zbyt kosztowny lub gdy jest nieznany. Wykorzystywany jest także dla problemów optymalizacyjnych, gdzie znajdowane jest bardzo szybko rozwiązanie, które nie jest idealne.

3.1 Ćwiczenie pierwsze

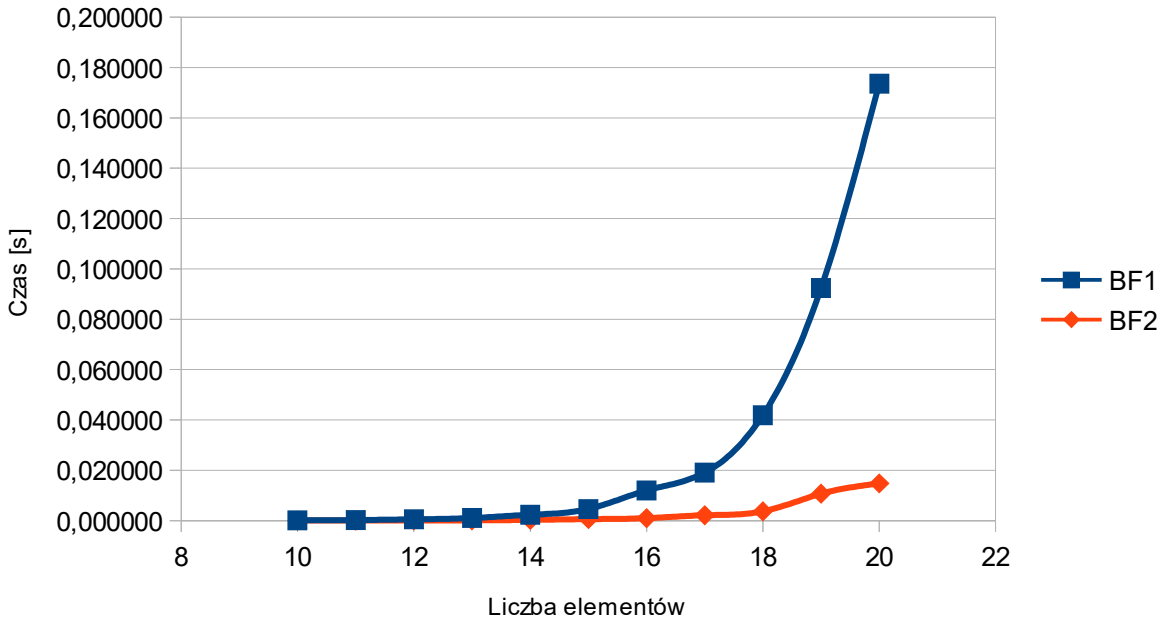
Ćwiczenie polegało na zbadaniu czasu trwania wyszukiwania najbardziej optymalnego rozwiązania dla metody programowania dynamicznego (PD), brutal force (BF1), brutal force z odcięciami (BF2) i metody heurystycznej dla największej zależności pomiędzy ciężarem, a wartością (GH4) w zależności od N i dla d = 0.5.

N	PD	BF1	BF2	GH4
10	0,000043	0,000130	0,000024	0,000002
11	0,000066	0,000234	0,000035	0,000002
12	0,000084	0,000511	0,000079	0,000002
13	0,000089	0,001045	0,000130	0,000002
14	0,000089	0,002341	0,000290	0,000002
15	0,000165	0,004614	0,000593	0,000002
16	0,000193	0,011990	0,000970	0,000002
17	0,000223	0,019044	0,002135	0,000003
18	0,000256	0,041864	0,003683	0,000004
19	0,000274	0,092431	0,010675	0,000004
20	0,000303	0,173620	0,014784	0,000004

Tabela 1: Tabela przedstawia czas trwania wyszukiwania rozwiązania dla wybranych algorytmów w zależności od liczby elementów (N) dla d = 0.5



Ilustracja 1: Wykres przedstawia czas trwania wyszukiwania rozwiązania dla programowania (PD) dynamicznego i metody heurystycznej (GH4) w zależności od liczby elementów (N) dla d = 0.5



Ilustracja 2: Wykres przedstawia czas trwania wyszukiwania rozwiązania dla brutal force (BF1) i brutal force z odcięciami (BF2) w zależności od liczby elementów (N) dla d = 0.5

3.2 Wnioski

Wersja decyzyjna problemu plecakowego znajduje się w klasie problemów NP – zupełnych, które nie są rozwiązywalne w czasie wielomianowym. Poprzez skorzystanie z programowanie dynamicznego okazuję się, że można otrzymać rozwiązanie w czasie pseudowielomianowym. Oznacza to, że klasa problemów trudnych nie jest jednorodna. Dzięki możliwości skorzystania z programowania dynamicznego problem plecakowy należy do klasy problemów NP – zupełnych w zwykłym sensie. Jeżeli dany problem nie ma rozwiązania pseudowielomianowego należy on do klasy problemów silnie NP – zupełnych.

Programowanie dynamiczne dzieli problem na zależne od siebie podproblemy, co pozwala na ponowne wykorzystanie wykonanych już wcześniej obliczeń. Także zastosowanie tego algorytmu będzie możliwe wtedy, gdy będzie możliwy podział na zależne od siebie podproblemy.

Złożoność obliczeniowa programowania dynamicznego wynosi O(n*b), gdzie n to liczba elementów, a b to pojemność. Algorytm brutal force jest algorytmem dokładnym o wykładniczej złożoności $O(2^n)$. Zaś jego modyfikacja z odcięciami jest szybsza, ale o podobnej złożoności. Algorytm heurystyczny ma złożoność równą metodzie sortowania, czyli w moim wypadku $O(n*\log_2 n)$.

Największą zaletą metody heurystycznej jest jej szybkość, nie mamy jednak pewności co do optymalnego rozwiązania. W metodzie programowania dynamicznego mamy pewność co do odpowiedzi, jednak największą wadą jest złożoność, która jest zależna od pojemności. Algorytm BF1 jest algorytmem najwolniejszym, który podaje poprawne rozwiązanie, jego modyfikacja, czyli BF2 jest szybsza i jest bardziej efektywna, gdy mamy stosunkowo niewielką liczbę paczek i dużą pojemność.

3.3 Ćwiczenie drugie

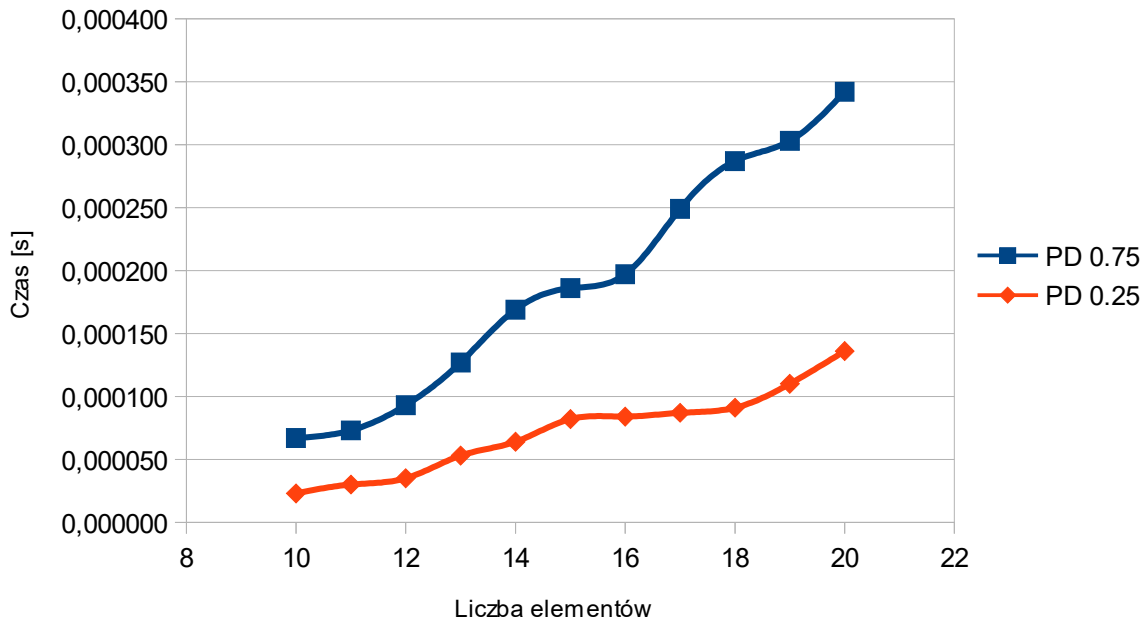
Ćwiczenie polegało na zbadaniu czasu trwania wyszukiwania najbardziej optymalnego rozwiązania dla metody programowania dynamicznego (PD), brutal force (BF1), brutal force z odcięciami (BF2) i metody heurystycznej dla największej zależności pomiędzy ciężarem, a wartością (GH4) w zależności od N i dla d = 0.25 i d = 0.75.

N	PD	BF1	BF2	GH4
10	0,000023	0,000109	0,000005	0,000002
11	0,000030	0,000218	0,000009	0,000002
12	0,000035	0,000488	0,000016	0,000002
13	0,000053	0,001332	0,000035	0,000002
14	0,000064	0,002122	0,000063	0,000002
15	0,000082	0,004878	0,000102	0,000002
16	0,000084	0,009474	0,000138	0,000002
17	0,000087	0,019404	0,000218	0,000002
18	0,000091	0,038249	0,000241	0,000003
19	0,000110	0,092178	0,000600	0,000003
20	0,000136	0,174646	0,000941	0,000003

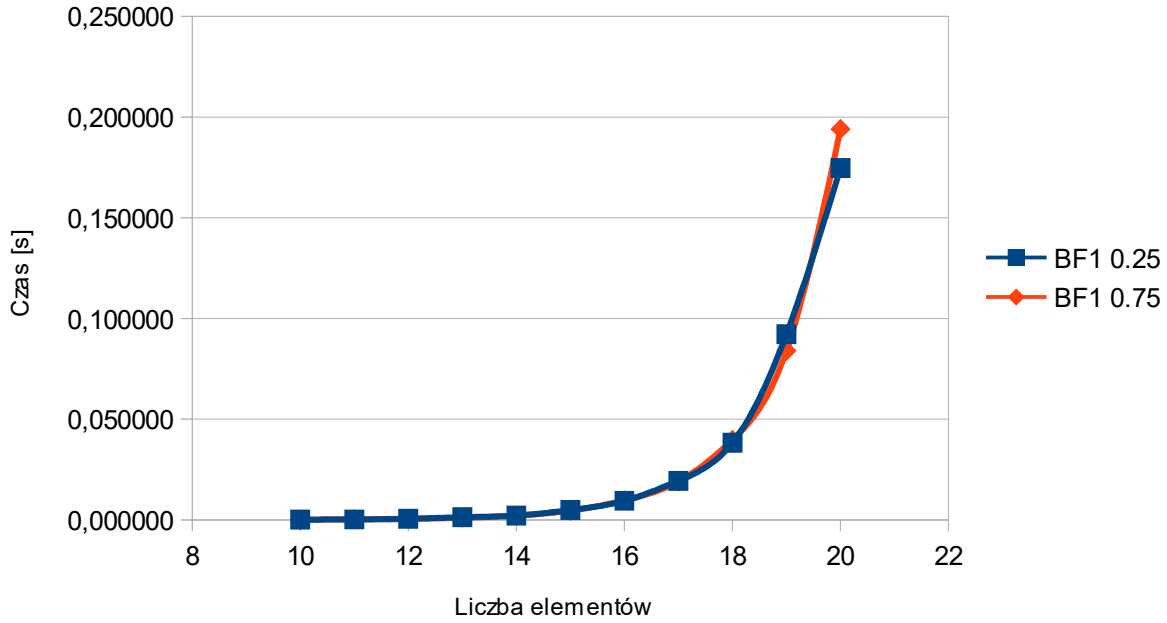
Tabela 2: Tabela przedstawia czas trwania wyszukiwania rozwiązania dla wybranych algorytmów w zależności od liczby elementów (N) dla d = 0.25

N	PD	BF1	BF2	GH4
10	0,000067	0,000110	0,000042	0,000002
11	0,000073	0,000235	0,000061	0,000002
12	0,000093	0,000517	0,000121	0,000002
13	0,000127	0,001008	0,000224	0,000002
14	0,000169	0,002159	0,000451	0,000003
15	0,000186	0,004554	0,000943	0,000003
16	0,000197	0,009635	0,001981	0,000003
17	0,000249	0,019059	0,003922	0,000003
18	0,000287	0,039700	0,007549	0,000003
19	0,000303	0,084012	0,015706	0,000003
20	0,000342	0,194009	0,037100	0,000003

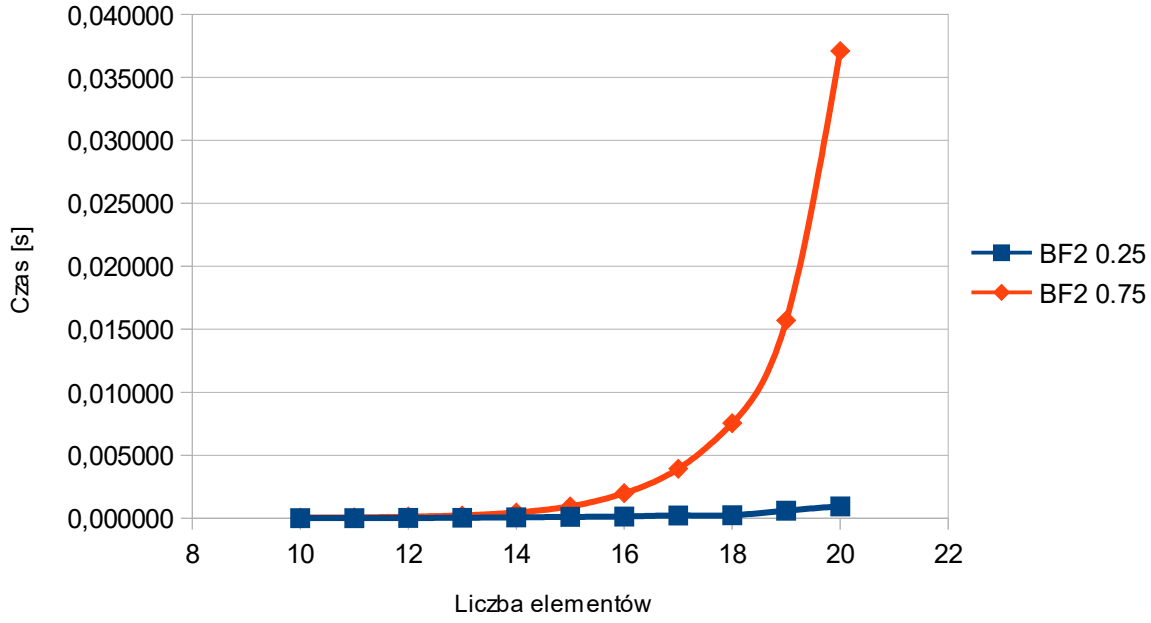
Tabela 3: Tabela przedstawia czas trwania wyszukiwania rozwiązania dla wybranych algorytmów w zależności od liczby elementów (N) dla d = 0.75



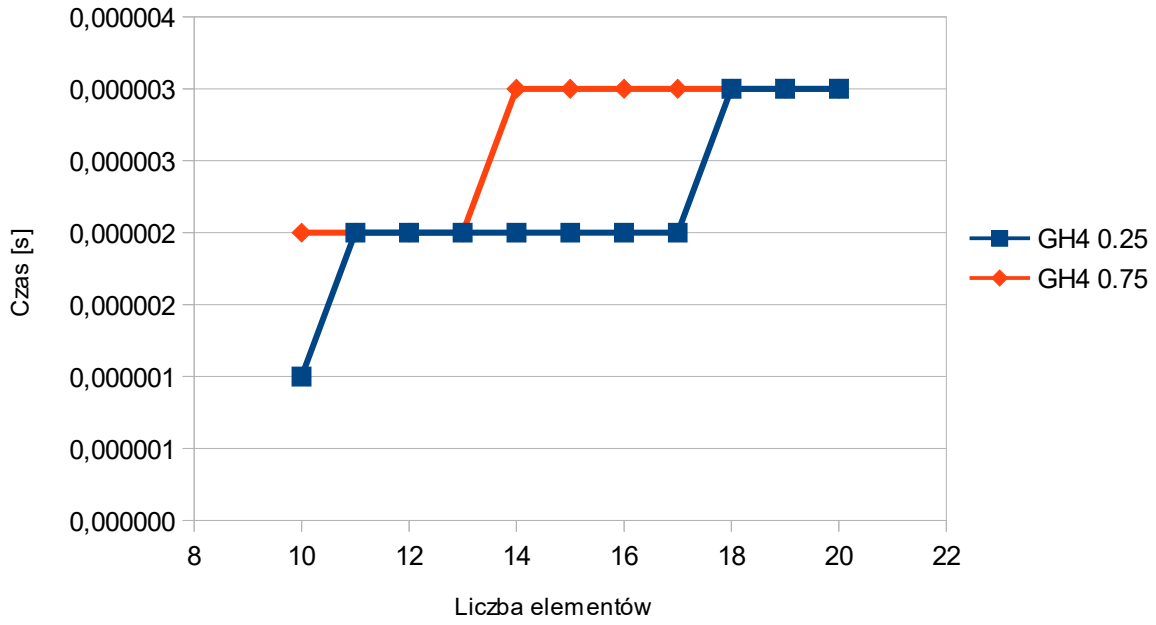
Ilustracja 3: Wykres przedstawia czas trwania wyszukiwania rozwiązania dla programowania dynamicznego od liczby elementów (N) dla d = 0.25 i d = 0.75



Ilustracja 4: Wykres przedstawia czas trwania wyszukiwania rozwiązania dla brutal force od liczby elementów (N) dla d = 0.25 i d = 0.75



Ilustracja 5: Wykres przedstawia czas trwania wyszukiwania rozwiązania dla brutal force z odcięciami od liczby elementów (N) dla d = 0.25 i d = 0.75



Ilustracja 6: Wykres przedstawia czas trwania wyszukiwania rozwiązania dla metody heurystycznej od liczby elementów (N) dla d = 0.25 i d = 0.75

3.4 Wnioski

Pojemność plecaka jest to waga wszystkich paczek przemnożona przez współczynnik d. Wyrażona jest wzorem b = d*Σs(ai).

We wszystkich przedstawionych metodach czas obliczeń dla poszczególnych metod zwiększał się wraz z ilością paczek. Różniło się zaś tempo wzrostu. Dla programowania dynamicznego była to zależność liniowa, dla heurystyki (GH4) liniowo logarytmiczna, a dla metody brutal force całkowitej i wraz z odcięciami była wykładnicza.

Dla metody brutal force całkowitej i heurystyki wartość pojemności jest nieważna. Różnice dla brutal force są małe, a dla heurystyki nieregularne, co jest spowodowane działaniem algorytmu sortowania.

W przypadku brutal force z odcięciami pojemność plecak ma już wpływ na czas wyszukiwania. Im mniejsza wartość b tym szybciej dana gałąź wyniku zostaje odrzucona. Wpływ pojemności możemy też zauważyć przy programowaniu dynamicznym, gdzie jego złożoność obliczeniowa jest zależna od pojemności. W tym wypadku im większa jest pojemność tym dłużej trwa wyszukiwanie.

3.5 Ćwiczenie trzecie

Ćwiczenie polegało na obliczeniu średniego błędu przy rozwiązaniach heurystycznych: losowej (GH1), minimum względem ciężaru (GH2), maksimum względem wartości (GH3) i maksymalną zależność pomiędzy wartością i masą (GH4). Dla minimum 10 wartości n w zależności od współczynnika d = 0.25, d = 0.5, d = 0.75.

N	100	200	300	400	500	600	700	800	900	1000
GH1	19,14%	19,43%	20,86%	17,78%	18,57%	18,18%	16,39%	19,06%	19,58%	17,77%
GH2	6,57%	8,61%	8,15%	8,39%	6,66%	7,02%	6,92%	7,35%	8,16%	7,59%
GH3	1,25%	1,27%	1,41%	2,32%	2,13%	2,03%	2,99%	1,65%	1,69%	1,90%
GH4	0,19%	0,10%	0,08%	0,03%	0,03%	0,01%	0,03%	0,02%	0,01%	0,05%

Tabela 4: Tabela przedstawia błąd dla metod heurystycznych w zależności od liczby elementów (N) dla d = 0.75

N	100	200	300	400	500	600	700	800	900	1000
GH1	32,29%	35,40%	31,61%	35,66%	32,28%	33,98%	34,75%	34,06%	30,02%	33,34%
GH2	8,43%	11,65%	10,20%	11,28%	10,56%	9,48%	10,33%	11,50%	10,72%	10,56%
GH3	7,48%	5,68%	5,27%	6,66%	8,16%	8,87%	6,95%	6,44%	7,16%	6,87%
GH4	0,21%	0,03%	0,01%	0,02%	0,02%	0,02%	0,02%	0,02%	0,01%	0,04%

Tabela 5: Tabela przedstawia błąd dla metod heurystycznych w zależności od liczby elementów (N) dla d = 0.5

N	100	200	300	400	500	600	700	800	900	1000
GH1	51,57%	42,71%	50,25%	51,29%	50,80%	48,34%	55,22%	51,13%	47,49%	51,38%
GH2	13,28%	10,75%	11,28%	12,65%	10,52%	12,43%	12,32%	13,43%	11,52%	11,98%
GH3	18,54%	24,65%	21,41%	19,71%	21,48%	20,34%	19,69%	23,58%	19,76%	21,36%
GH4	0,41%	0,13%	0,12%	0,11%	0,05%	0,01%	0,02%	0,03%	0,01%	0,09%

Tabela 6: Tabela przedstawia błąd dla metod heurystycznych w zależności od liczby elementów (N) dla d = 0.25

Błąd średni	GH1	GH2	GH3	GH4
d = 0.25	50,02%	11,98%	21,36%	0,09%
d = 0.5	33,34%	10,56%	6,87%	0,04%
d = 0.75	18,68%	7,59%	1,90%	0,05%
Średnia	34,01%	10,04%	10,04%	0,06%

Tabela 7: Tabela przedstawia średni błąd dla metod heurystycznych w zależności współczynnika d

3.6 Wnioski

Spośród zaimplementowanych metod GH wszystkie są heurystycznymi. Heurystyczne, czyli metody znajdowania rozwiązań, które nie gwarantują ich poprawności. Prócz GH1 (losowego) wszystkie są one zachłanne i listowe. Algorytm zachłanny to taki, który wybiera najlepsze rozwiązanie w danym momencie. Nie analizuje, czy w kolejnych krokach jest sens wykonywać

dane działanie. Algorytm listowny to taki, który wpierw sortuje pewne dane, a potem odczytuje je według kolejności.

Z ostatniej tabeli średnich możemy odczytać, że im większa pojemność tym algorytmy heurystyczne są bardziej poprawne. Największą różnicę pokazuje GH1 (losowe wybieranie paczek), zaś algorytmy GH2 (minimalny ciężar) i GH3 (maksymalna wartość) są bardzo podobne względem błędów. Dla GH2 mogą zostać wybrane małe paczki o znikomej wartości, które mogłyby być zastąpione jedną dużą o większej wartości, zaś dla GH3 sytuacja jest analogiczna. Może zostać wybrana duża paczka o dużej wartości, lecz lepsze na jej miejsce byłyby mniejsze o sumarycznie większej wartości. GH4 (stosunek ciężaru do wartości) jest opcją najkorzystniejszą, która bierze pod uwagę ciężar jak i masę, jednak nie gwarantuje najlepszego rozwiązania. Im większa jest pojemność tym wyniki są dokładniejsze. Wraz z wzrostem liczby elementów dokładność GH4 i GH2 wzrasta, a GH3 maleje.

Metodę zachłanną możemy wykorzystać nie tylko do problemu plecakowego, ale także innych. W problemie komiwojażera algorytm zachłanny może wybierać najbliższe nieodwiedzone miasta.

4. Bibliografia

- Wiadomości podane na ćwiczeniach z Algorytmów i Struktur Danych
- Informacje ze strony: (http://iair.mchtr.pw.edu.pl/~bputz/aisd_cpp/lekcja2/segment5/main.htm) dostęp z dnia 28.05.2017 r.
- Informacje ze strony: (http://www.algorytm.org/kurs-algorytmiki/programowanie-dynamiczne.html) dostęp z dnia 28.05.2017 r.
- Informacje ze strony: (http://edu.pjwstk.edu.pl/wyklady/nai/scb/wyklad5/w5.htm) dostęp z dnia 28.05.2017 r.