Sztuczna Inteligencja i inżynieria wiedzy

Sprawozdanie z zadania drugiego laboratorium

„Problemy spełniania ograniczeń”

Prowadzący laboratorium:
dr. inż. Paweł Myszkowski

Przygotował:
Aleksander Poławski

1. Wstęp

Celem zadania drugiego była implementacja algorytmów przeszukiwania wstecz i w przód do rozwiązywania problemów CSP.

Algorytmy należało zaimplementować w taki sposób, aby były w stanie rozwiązać problemy łamigłówkowe „Sudoku” oraz „Jolka”.

Dodatkowo należało zbadać wpływ różnych heurystyk doboru zmiennej i wartości w następnym kroku algorytmów.

Program zawierający wszystkie wymienione składowe napisano w języku C++, aby uzyskać jak najlepszą wydajność.

2. Opracowane algorytmy i wstępne badania:

W celu zrealizowania zadania zaimplementowano algorytm:
- przeszukiwania z nawrotami (dalej nazywany „backtracking”)
- sprawdzania w przód (dalej nazywany „forwardChecking”)

Dodatkowo zaimplementowano dwie heurystyki wyboru następnej zmiennej:
- w kolejności deklaracji (dalej nazywanej „as declared”)
- wybór zmiennej najbardziej ograniczonej (dalej nazywanej „most constrained”)

oraz dwie heurystyki wyboru następnej wartości z dziedziny:
- w kolejności deklaracji (dalej nazywanej „as declared”)
- losowo (dalej nazywanej „random”)

Przeprowadzono badania dla przykładowego zadania Sudoku o trudności = 5.0 i id = 27:

Próba	**Ustawienia**		**Nawroty**	**Nawroty do pierwszego rozwiązania**	**Odwiedzone węzły**	**Odwiedzone węzły do pierwszego rozwiązania**	**Czas wykonania**	**Liczba rozwiązań**
#1	**Algorytm**	backtracking	68 113	14 288	613 017	128 868	3.34 s	1
	Wybór zmiennej	as declared						
	Wybór wartości	as declared						
#2	**Algorytm**	backtracking	68 113	52 111	613 017	469 301	3.39 s	1
	Wybór zmiennej	as declared						
	Wybór wartości	random						

#3	**Algorytm**	backtracking	35 265	12 586	317 385	113 550	8.70 s	1
	Wybór zmiennej	most constrained						
	Wybór wartości	as declared						
#4	**Algorytm**	backtracking	35 265	29 410	317 385	264 978	9.25 s	1
	Wybór zmiennej	most constrained						
	Wybór wartości	random						
#5	**Algorytm**	forwardChecking	67 721	14 265	229 441	50 738	6.72 s	1
	Wybór zmiennej	as declared						
	Wybór wartości	as declared						
#6	**Algorytm**	forwardChecking	67 721	56 191	229 441	187 874	7.02 s	1
	Wybór zmiennej	as declared						
	Wybór wartości	random						
#7	**Algorytm**	forwardChecking	34 631	12 409	124 032	46 060	10.48 s	1
	Wybór zmiennej	most constrained						
	Wybór wartości	as declared						
#8	**Algorytm**	forwardChecking	34 631	22 462	124 032	79 014	10.34 s	1
	Wybór zmiennej	most constrained						
	Wybór wartości	random						

Wnioski - Sudoku:

- zmiana heurystyki wyboru wartości z kolejności deklaracji na losową nie wpływa na całkowitą liczbę wykonanych nawrotów i odwiedzonych węzłów. Zmienia się natomiast losowo liczba węzłów i nawrotów do znalezienia pierwszego rozwiązania.

- zmiana heurystyki wyboru zmiennej z kolejności deklaracji na dobór węzła najbardziej ograniczonego znacznie zmniejsza całkowitą liczbę nawrotów i odwiedzonych węzłów. Wpływa jednak negatywnie na czas wykonania programu.

- algorytm „backtracking” jest pod względem liczby wykonanych nawrotów i odwiedzonych węzłów nieznacznie gorszy od algorytmu „forwardChecking”. Jest on jednak szybszy pod względem czasu wykonania programu.

- najlepszym rozwiązaniem jest wybór heurystyki zmiennej „most constrained”, heurystyki wartości „as declared” i algorytmu „forwardChecking”

Przeprowadzono badania dla przykładowego zadania „Jolka” o id = 1:

Próba	Ustawienia		Nawroty	Nawroty do pierwszego rozwiązania	Odwiedzone węzły	Odwiedzone węzły do pierwszego rozwiązania	Czas wykonania	Liczba rozwiązań
#1	**Algorytm**	backtracking	2 880 217	78 506	12 534 640	342 647	104.63s	2
	Wybór zmiennej	as declared						
	Wybór wartości	as declared						
#2	**Algorytm**	backtracking	2 880 217	135 790	12 534 640	593 162	121.47s	2
	Wybór zmiennej	as declared						
	Wybór wartości	random						
#3	**Algorytm**	backtracking	56	0	250	59	0.00s	2
	Wybór zmiennej	most constrained						
	Wybór wartości	as declared						
#4	**Algorytm**	backtracking	45	3	250	88	0.00s	2
	Wybór zmiennej	most constrained						
	Wybór wartości	random						
#5	**Algorytm**	forwardChecking	114	22	114	40	0.01s	2
	Wybór zmiennej	as declared						
	Wybór wartości	as declared						
#6	**Algorytm**	forwardChecking	114	23	114	41	0.01s	2
	Wybór zmiennej	as declared						
	Wybór wartości	random						
#7	**Algorytm**	forwardChecking	45	0	45	18	0.01s	2
	Wybór zmiennej	most constrained						
	Wybór wartości	as declared						

#8	**Algorytm**	forwardChecking	45	1	45	19	0.01s	2
	Wybór zmiennej	most constrained						
	Wybór wartości	random						

Wnioski - Jolka:

- podobnie jak w przypadku zadań typu „Sudoku” algorytm „backtracking” jest pod względem liczby wykonanych nawrotów i odwiedzonych węzłów nieznacznie gorszy od algorytmu „forwardChecking”.

- podobnie jak w przypadku zadań typu „Sudoku” zmiana heurystyki wyboru wartości z kolejności deklaracji na losową nie wpływa na całkowitą liczbę wykonanych nawrotów i odwiedzonych węzłów. Zmienia się natomiast losowo liczba węzłów i nawrotów do znalezienia pierwszego rozwiązania.

- heurystyka doboru zmiennej na podstawie kolejności deklaracji jest w przypadku algorytmu „backtracking” zupełnie nieefektywna. W przypadku użycia jej w kombinacji z algorytmem „forwardChecking” staje się użyteczna. Heurystyka wyboru zmiennej na podstawie ilości ograniczeń jest jednak lepsza w każdym przypadku.

- podobnie jak w przypadku zadań typu „Sudoku” najlepszym rozwiązaniem jest wybór heurystyki zmiennej „most constrained”, heurystyki wartości „as declared” i algorytmu „forwardChecking” .

3. Badania dla pozostałych instancji łamigłówek „Jolka”

Postanowiono pominąć badania z heurystyką wyboru zmiennej na podstawie kolejności deklaracji z powodu jej skrajnej nieefektywności.

Przeprowadzono badania dla zadania „Jolka” o id = 0:

Próba	**Ustawienia**		**Nawroty**	**Nawroty do pierwszego rozwiązania**	**Odwiedzone węzły**	**Odwiedzone węzły do pierwszego rozwiązania**	**Czas wykonania**	**Liczba rozwiązań**
#1	**Algorytm**	backtracking	8	0	28	13	0.00s	1
	Wybór zmiennej	most constrained						
	Wybór wartości	as declared						
#2	**Algorytm**	backtracking	8	0	28	13	0.00s	1
	Wybór zmiennej	most constrained						
	Wybór wartości	random						

#3	**Algorytm**	forwardChecking	8	0	8	6	0.00s	1
	Wybór zmiennej	most constrained						
	Wybór wartości	as declared						
#4	**Algorytm**	forwardChecking	8	1	8	7	0.00s	1
	Wybór zmiennej	most constrained						
	Wybór wartości	random						

Przeprowadzono badania dla zadania „Jolka” o id = 2:

Próba	**Ustawienia**		**Nawroty**	**Nawroty do pierwszego rozwiązania**	**Odwiedzone węzły**	**Odwiedzone węzły do pierwszego rozwiązania**	**Czas wykonania**	**Liczba rozwiązań**
#1	**Algorytm**	backtracking	89	37	577	348	0.01s	1
	Wybór zmiennej	most constrained						
	Wybór wartości	as declared						
#2	**Algorytm**	backtracking	89	25	577	284	0.01s	1
	Wybór zmiennej	most constrained						
	Wybór wartości	random						
#3	**Algorytm**	forwardChecking	56	15	56	47	0.01s	1
	Wybór zmiennej	most constrained						
	Wybór wartości	as declared						
#4	**Algorytm**	forwardChecking	56	9	56	41	0.01s	1
	Wybór zmiennej	most constrained						
	Wybór wartości	random						

Przeprowadzono badania dla zadania „Jolka” o id = 3:

Próba	Ustawienia		Nawroty	Nawroty do pierwszego rozwiązania	Odwiedzone węzły	Odwiedzone węzły do pierwszego rozwiązania	Czas wykonania	Liczba rozwiązań
#1	**Algorytm**	backtracking	353	162	9 944	5 996	0.11s	1
	Wybór zmiennej	most constrained						
	Wybór wartości	as declared						
#2	**Algorytm**	backtracking	353	188	9 944	6 697	0.11s	1
	Wybór zmiennej	most constrained						
	Wybór wartości	random						
#3	**Algorytm**	forwardChecking	201	53	201	137	0.22s	1
	Wybór zmiennej	most constrained						
	Wybór wartości	as declared						
#4	**Algorytm**	forwardChecking	201	52	201	136	0.23s	1
	Wybór zmiennej	most constrained						
	Wybór wartości	random						

Przeprowadzono badania dla zadania „Jolka” o id = 4:

Próba	Ustawienia		Nawroty	Nawroty do pierwszego rozwiązania	Odwiedzone węzły	Odwiedzone węzły do pierwszego rozwiązania	Czas wykonania	Liczba rozwiązań
#1	**Algorytm**	backtracking	130	2	985	545	0.04s	1
	Wybór zmiennej	most constrained						
	Wybór wartości	as declared						
#2	**Algorytm**	backtracking	130	5	985	567	0.04s	1
	Wybór zmiennej	most constrained						
	Wybór wartości	random						

#3	**Algorytm**	forwardChecking	128	2	128	125	0.06s	1
	Wybór zmiennej	most constrained						
	Wybór wartości	as declared						
#4	**Algorytm**	forwardChecking	128	4	128	127	0.07s	1
	Wybór zmiennej	most constrained						
	Wybór wartości	random						

Wnioski:

- badania pozostałych plików łamigłówek „Jolka” potwierdzają wnioski wysunięte we wstępnych badaniach

4. Badania dla pozostałych instancji łamigłówek „Sudoku”

Przeprowadzono badania dla przykładowego zadania Sudoku o trudności = 2.0 i id = 12:

Próba	**Ustawienia**		**Nawroty**	**Nawroty do pierwszego rozwiązania**	**Odwiedzone węzły**	**Odwiedzone węzły do pierwszego rozwiązania**	**Czas wykonania**	**Liczba rozwiązań**
#1	**Algorytm**	backtracking	41 826	16 770	376 434	151 203	2.09 s	1
	Wybór zmiennej	as declared						
	Wybór wartości	as declared						
#2	**Algorytm**	backtracking	41 826	27 116	376 434	244 341	2.12 s	1
	Wybór zmiennej	as declared						
	Wybór wartości	random						
#3	**Algorytm**	backtracking	9 845	7 209	88 605	65 154	2.14 s	1
	Wybór zmiennej	most constrained						
	Wybór wartości	as declared						
#4	**Algorytm**	backtracking	9 845	3 211	88 605	29 145	2.04 s	1
	Wybór zmiennej	most constrained						
	Wybór wartości	random						

#5	**Algorytm**	forwardChecking	41 713	16 752	143 397	57 664	4.36 s	1
	Wybór zmiennej	as declared						
	Wybór wartości	as declared						
#6	**Algorytm**	forwardChecking	41 713	3 177	143 397	10 613	4.59 s	1
	Wybór zmiennej	as declared						
	Wybór wartości	random						
#7	**Algorytm**	forwardChecking	9 825	7 200	33 052	24 548	2.37 s	1
	Wybór zmiennej	most constrained						
	Wybór wartości	as declared						
#8	**Algorytm**	forwardChecking	9 825	8 818	33 052	29 334	2.41 s	1
	Wybór zmiennej	most constrained						
	Wybór wartości	random						

Przeprowadzono badania dla przykładowego zadania Sudoku o trudności = 6.0 i id = 33:

Próba	**Ustawienia**		**Nawroty**	**Nawroty do pierwszego rozwiązania**	**Odwiedzone węzły**	**Odwiedzone węzły do pierwszego rozwiązania**	**Czas wykonania**	**Liczba rozwiązań**
#1	**Algorytm**	backtracking	804 738	219 059	7 242 642	1 971 809	36.69s	1
	Wybór zmiennej	as declared						
	Wybór wartości	as declared						
#2	**Algorytm**	backtracking	804 738	626 012	7 242 642	5 634 417	40.86s	1
	Wybór zmiennej	as declared						
	Wybór wartości	random						
#3	**Algorytm**	backtracking	136 392	96 651	1 227 528	870 137	28.78s	1
	Wybór zmiennej	most constrained						
	Wybór wartości	as declared						

#4	**Algorytm**	backtracking	136 392	99 851	1 227 528	898 978	29.39s	1
	Wybór zmiennej	most constrained						
	Wybór wartości	random						
#5	**Algorytm**	forwardChecking	787 846	214 454	2 229 737	604 531	67.11s	1
	Wybór zmiennej	as declared						
	Wybór wartości	as declared						
#6	**Algorytm**	forwardChecking	787 846	136 259	2 229 737	395 123	69.62s	1
	Wybór zmiennej	as declared						
	Wybór wartości	random						
#7	**Algorytm**	forwardChecking	130 443	92 464	401 072	285 450	33.90s	1
	Wybór zmiennej	most constrained						
	Wybór wartości	as declared						
#8	**Algorytm**	forwardChecking	130 443	33 873	401 072	104 407	34.38s	1
	Wybór zmiennej	most constrained						
	Wybór wartości	random						

Przeprowadzono badania dla przykładowego zadania Sudoku o trudności = 8.0 i id = 42:

Próba	**Ustawienia**		**Nawroty**	**Nawroty do pierwszego rozwiązania**	**Odwiedzone węzły**	**Odwiedzone węzły do pierwszego rozwiązania**	**Czas wykonania**	**Liczba rozwiązań**
#1	**Algorytm**	backtracking	97 274	5 482	875 115	49 629	4.63s	40
	Wybór zmiennej	as declared						
	Wybór wartości	as declared						
#2	**Algorytm**	backtracking	97 274	4 678	875 115	42 386	5.06s	40
	Wybór zmiennej	as declared						
	Wybór wartości	random						

#3	**Algorytm**	backtracking	197 882	5 151	1 780 587	46 650	35.73s	40
	Wybór zmiennej	most constrained						
	Wybór wartości	as declared						
#4	**Algorytm**	backtracking	197 882	19 585	1 780 587	176 550	36.82s	40
	Wybór zmiennej	most constrained						
	Wybór wartości	random						
#5	**Algorytm**	forwardChecking	96 804	5 440	247 177	13 993	7.99s	40
	Wybór zmiennej	as declared						
	Wybór wartości	as declared						
#6	**Algorytm**	forwardChecking	96 804	6 083	247 177	17 809	8.80s	40
	Wybór zmiennej	as declared						
	Wybór wartości	random						
#7	**Algorytm**	forwardChecking	196 149	5 151	474 348	12 650	41.67s	40
	Wybór zmiennej	most constrained						
	Wybór wartości	as declared						
#8	**Algorytm**	forwardChecking	196 149	4 849	474 348	14 836	41.38s	40
	Wybór zmiennej	most constrained						
	Wybór wartości	random						

Wnioski:

- badania pozostałych plików łamigłówek „Sudoku” potwierdzają wnioski wysunięte we wstępnych badaniach. Wyjątkiem był problem o id=42, gdzie rozwiązaniem jest aż 40 układów. W tym wypadku heurystyka dobierająca zmienną na podstawie najbardziej ograniczonej stała się mało wydajna.

5. Podsumowanie

Algorytmy przeszukiwania wstecz i w przód pozwalają znaleźć rozwiązania problemów CSP takich jak łamigłówki „Sudoku” i „Jolka”. Aby działały prawidłowo wymagają one jednak doprecyzowania, na przykład za pomocą heurystyk wyboru kolejnych zmiennych/wartości.

Pomyślnie zrealizowano wszystkie wytyczne zadania oraz dokonano odpowiednich badań.