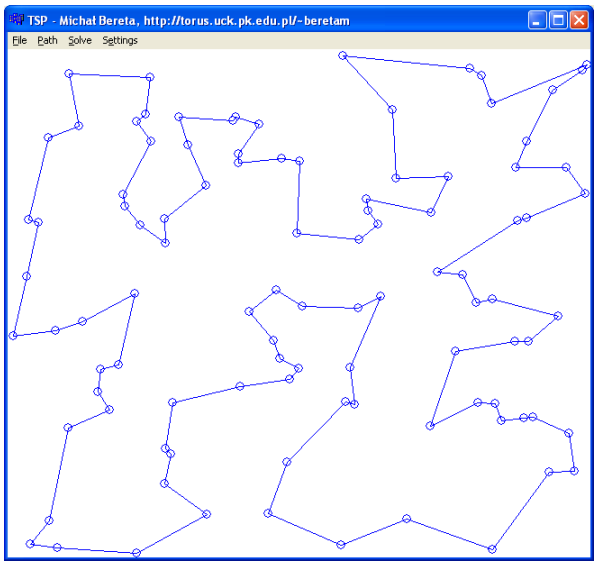
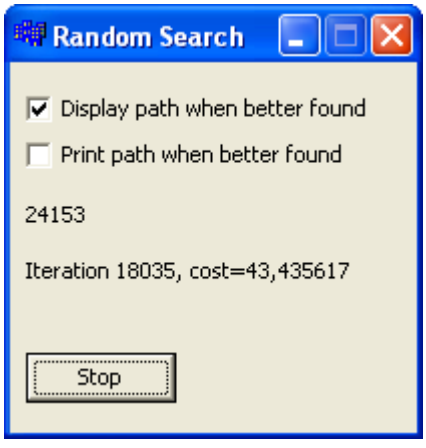
Problem komiwojażera
(Traveling Salesman Problem - TSP)

Michał Bereta, beretam@torus.uck.pk.edu.pl



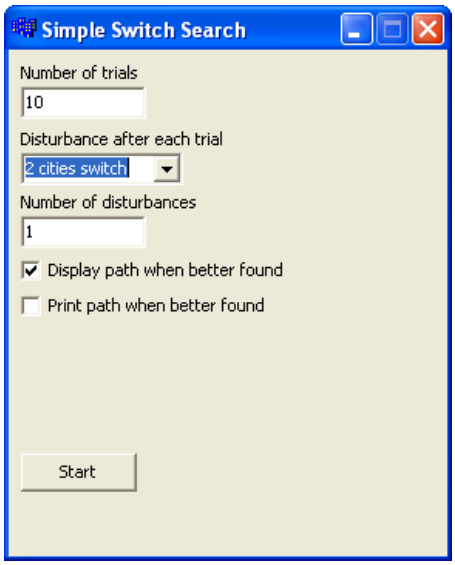
Program pozwala przetestować kilka heurystyk, które mogą być wykorzystywane do rozwiązywania problemu komiwojażera.

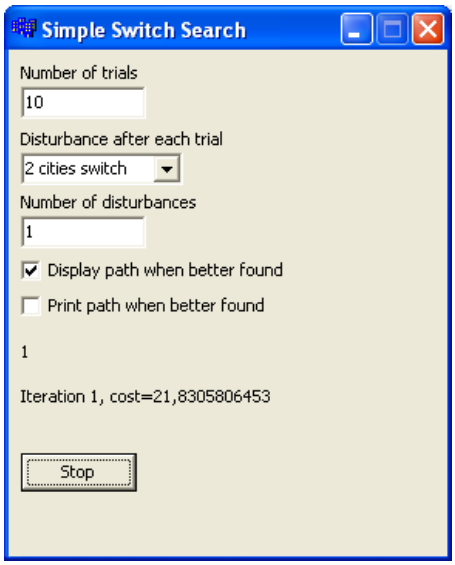
1. **Przeszukiwanie losowe (Solve → Random Search F1)**. Jest to bardzo prosta heurystyka. W każdej iteracji generowana jest losowa trasa. Jest ona akceptowana wtedy, gdy jest lepsza niż dotychczasowe najlepsze rozwiązanie. Jest to podejście bardzo mało wydajne i przedstawiony jedynie po to by pokazać jego nieadekwatność.



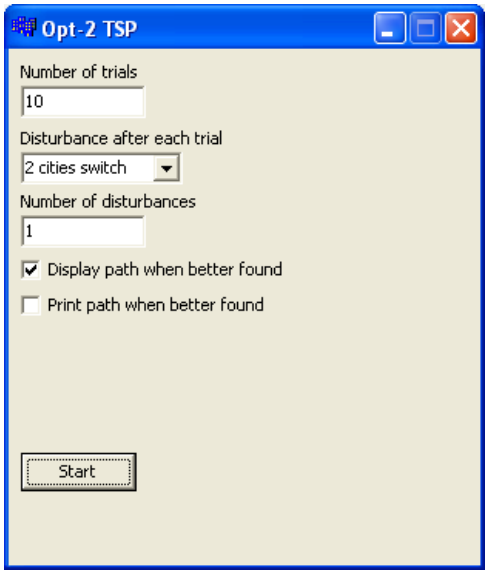
2. **Proste przestawianie miast na trasie (Solve → Simple Switch Search F2).** W podejściu tym wykorzystujemy dotychczasowe najlepsze rozwiązanie w celu wygenerowania lepszego. W tym celu przestawiamy dowolną parę miast na trasie, jeśli tylko prowadzi to do lepszego rozwiązania. Oczywiście, przeważnie będzie to jedynie minimum lokalne. Jeśli przestawienie żadnej pary miast nie daje poprawy, algorytm wprowadza losowe zaburzenie do rozwiązania (**Disturbance after each trial).** Zaburzeniem tym jest również przestawienie dwóch wylosowanych miast, jednakże jest ono akceptowane nawet wtedy, gdy pogarsza trasę. Liczbą takich losowych zaburzeń można sterować za pomocą parametru ***Number of disturbances***. Po

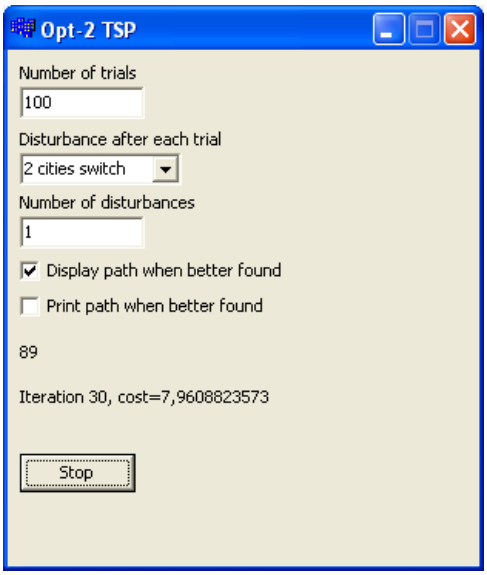
wprowadzeniu zaburzenia ponownie sprawdza się, czy przestawienie jakiejkolwiek pary miast daje lepsze rozwiązanie. Całą tę procedurę powtarza się tyle razy ile wynosi parametr ***Number of trials***.



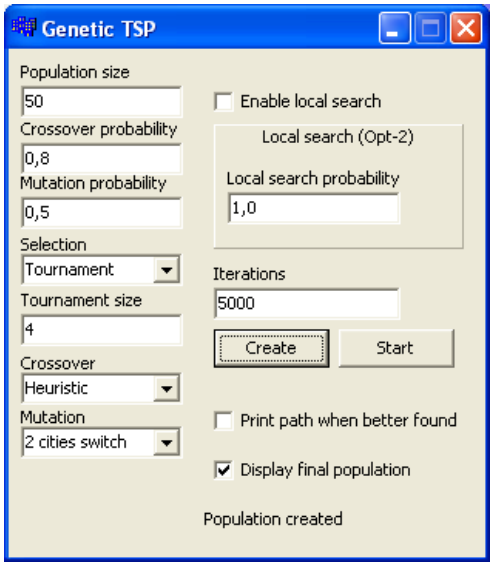


3. **Metoda najbliższego sąsiada – nearest neighbor (Solve → NN F3)**. W heurystyce tej wybiera się miasto początkowe a następnie przechodzi z niego do tego miast, które jest najbliżej, zapisując je na liście odwiedzonych miast. Z każdego kolejnego miasta przechodzi się do najbliższego, które jeszcze nie zostało odwiedzone. Całą procedurę powtarza się dla każdego miasta jako miasta początkowego, gdyż ma to wpływ na znalezione rozwiązanie. Najlepsze spośród rozwiązań dla każdego z miast jako początkowego jest wynikiem działania całego algorytmu. W heurystyce tej liczymy na to, iż wykonując lokalnie najlepsze wybory, otrzymane rozwiązanie również będzie dobrej jakości. Podejście to daje wyniki dalekie od optymalnego, lecz ze względu na swoją szybkość może służyć jako pierwszy krok przed wykorzystaniem bardziej zaawansowanych heurystyk, np. algorytmu poszukiwań rozwiązań 2-optymalnych, omówionego w następnym punkcie. Warto jednak podkreślić, iż metoda NN jest procedurą deterministyczną, tzn. za każdym razem da ten sam wynik.
4. **Algorytm 2-optymalny (Solve → Opt2 F4)**. W podejściu tym bazujemy na obserwacji, iż krzyżujące się połączenia między miastami są zawsze gorsze niż takie, które się nie krzyżują. W algorytmie tym zatem sprawdza się wszystkie możliwe pary krawędzi i jeśli którakolwiek zawiera krawędzie krzyżujące się, następuje takie przestawienie czterech miast na trasie, by krzyżujące się krawędzie zostały zastąpione przez takie, które się nie krzyżują. Jednakże, brak krzyżujących się krawędzi wcale nie gwarantuje optymalności rozwiązania i cały proces przeważnie kończy się w minimum lokalnym. Aby „uciec” z tego minimum lokalnego wprowadzić można losowe zaburzenia do aktualnie najlepszej trasy (***Disturbance after each trial***), poprzez losowe przestawienie miast na trasie (***2 cities switch***), bez względu na to czy poprawia to rozwiązanie czy też nie. Po takim losowym zaburzeniu następuje ponowna próba usunięcia krzyżujących się krawędzi. Cała procedura jest powtarzana przez zadaną liczbę iteracji bądź też dopóki prowadzi do poprawy rozwiązania. W programie istnieje możliwość ustawienia liczby losowych zaburzeń (***Number of disturbances***) jak również liczby powtórzeń całej procedury (***Number of trials***).





5. Algorytm genetyczny **(Solve → Genetic TSP F5)**. Algorytmy genetyczne od dawna są stosowane do rozwiązywania problemu komiwojażera. Sposób ich zastosowania w problemie TSP nie jest jednak oczywisty. Przykładowo, forma reprezentacji osobnika kodującego rozwiązanie, czyli trasę komiwojażera, nie jest jednoznaczna. Reprezentacja osobnika w populacji jako listy kolejno uporządkowanych miast nie jest jedyną możliwą formą reprezentacji rozwiązania. Idąc krok dalej w tych rozważaniach, od formy reprezentacji osobnika zależy postać zastosowanych operatorów genetycznych (krzyżowania i mutacji). Program umożliwia przetestowanie działanie algorytmu genetycznego z najbardziej typowymi operatorami krzyżowania (krzyżowanie heurystyczne i OX) oraz mutacji (losowe przestawienie dwóch miast na trasie). Do wyboru są dwie metody selekcji, choć preferowaną jest selekcja turniejowa. Zaobserwować można bardzo duży wpływ rozmiaru turnieju na jakość działania algorytmu genetycznego.



Heurystyki można ze sobą łączyć. Przykładem jest zastosowanie najpierw metody NN a następnie Opt-2. Bardzo ciekawą heurystyką jest połączenie algorytmu genetycznego z lokalnym przeszukiwaniem. W tym podejściu każdy osobnik w populacji, zanim zostanie poddany selekcji, poddawany jest „ulepszaniu" z wykorzystaniem pewnej metody przeszukiwania lokalnego, takiego jak algorytm 2-optymalny. Program umożliwia przetestowanie tego podejścia poprzez zaznaczenie „**Enable local search**" na panelu sterowania algorytmu genetycznego. W tym podejściu konieczne jest uruchomienie algorytmu Opt2 dla każdego osobnika z osobna (z pewnym prawdopodobieństwem), jest to zatem bardzo kosztowna obliczeniowo procedura. Najdłużej trwa pierwsza generacja - osobniki są losowe, więc algorytm Opt2 trwa długo. Osobniki 2-optymalne w wyniku krzyżowania i mutacji przestają być 2-optymalne, dlatego w każdej kolejnej generacji przed krokiem selekcji poddawane są one z pewnym prawdopodobieństwem lokalnej optymalizacji algorytmem Opt2. Można zaobserwować, iż takie podejście daje wyniki lepsze niż zastosowanie osobno algorytmu Opt2 czy też osobno algorytmu genetycznego. Zapłacić za to trzeba wydłużonym czasem obliczeń.

UWAGA! Po wprowadzenie jakiejkolwiek zmiany w parametrach algorytmu genetycznego należy przycisnąć przycisk „Create" w celu stworzenia nowej populacji na podstawie nowych parametrów.

UWAGA! Po naciśnięciu przycisku „Stop" w przypadku jakiegokolwiek algorytmu, należy zaczekać na rzeczywiste zakończenie obliczeń przed ponownym przyciśnięciem przycisku „Start".

Bardziej szczegółową dyskusję na temat zastosowania algorytmu genetycznego do problemu komiwojażera oraz dokładny opis zastosowanych w programie operatorów genetycznych znaleźć można w rozdziale 10-tym książki:

Zbigniew Michalewicz „Algorytmy genetyczne + struktury danych = programy ewolucyjne", WNT, Warszawa 2003.