Wrocław, 28 czerwca 2021

Zaawansowane **T**echniki **O**ptymalizacji

Skrócony poradnik prowadzenia badań

prowadzący: *dr inż. Jarosław Rudy*

1 Wstęp

Poniższy dokument zawiera opis niektórych zagadnień istotnych podczas procesu dobierania parametrów algorytmów, przeprowadzania badań numerycznych i pomiarów, rysowania wykresów i wyciągania wniosków na podstawie danych. Ma on na celu ułatwić powyższe czynności i pozwolić na uniknięcie typowych błędów popełnianych w ich trakcie. Pomimo że poniższy poradnik liczy kilkanaście stron, jest on wciąż skrócony – niektóre zagadnienia traktuje w sposób uproszczony, a niektóre zakłada że są znane czytelnikowi. Jednocześnie część z opisywanych zagadnień jest dość zaawansowana lub wymaga znacznego zwiększenia nakładu pracy podczas badań. W związku z tym zrozumiałym jest, że nie zawsze uda się poprawnie i w pełni zastosować opisane tutaj metody i wskazówki. Dlatego drugim celem poradnika jest przedstawienie jak pewne elementy procesu badawczego mogą wpływać na uzyskane wyniki i wnioski i powodować że są one błędne lub mało wiarygodne.

2 Dobór punktów danych

Przy próbie ustalenia zależności dwóch wielkości poprzez uruchomienie i pomiar algorytmu, jedną z pierwszych decyzji jest wybór punktów pomiarowych tzn. wartości na osi X da których dokonujemy pomiaru. Wybór zbyt wielu punktów nie jest błędem, ale zwielokrotnia czas potrzebny na wykonanie badań. Dlatego najczęściej wybiera się mniejszą liczbę punktów, lecz poprawny dobór liczby i rozmieszczenia takich punktów nie jest zawsze oczywisty.

Zasadnicze cele są dwa: uzyskanie możliwie jednorodnego zbioru punktów oraz odpowiednio dużej liczby punktów. Złym pomysłem jest mniej niż 4 punkty pomiarowe, gdyż bardzo trudno jest wtedy zobaczyć (lub wydedukować) realny kształt funkcji na wykresie. Czasami pewien przedział osi X chcemy zbadać bardziej dokładnie, więc naturalnie punkty pomiarowe są tam umieszczane gęściej. Takie zagęszczenia nie są zwykle problemem o ile pozostałe fragmenty nie są zbyt rzadkie.

Najczęściej spotyka się jedną z dwóch metod doboru punktów:

1. Ciąg arytmetyczny, gdzie współrzędna x wzrasta o stały czynnik np.:

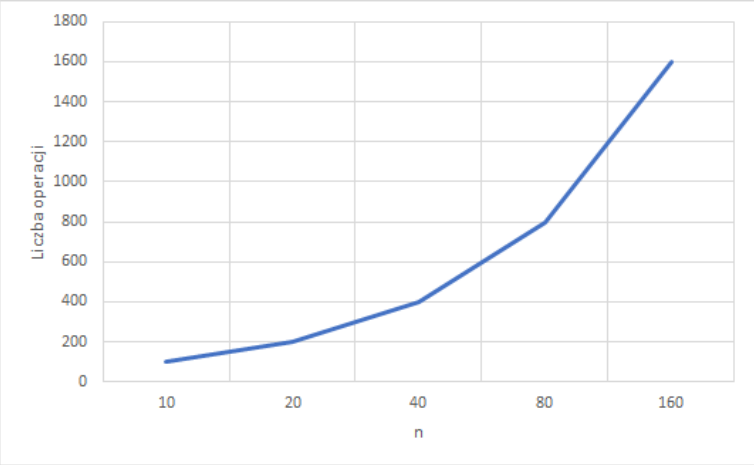
$$n \in \{10, 15, 20, 25, 30, 35, 40, 45, 50\}$$

2. Ciąg geometryczny, gdzie współrzędna x jest mnożona o stały czynnik np.

$$n \in \{10, 20, 40, 80, 160, 320\}$$

Dobór geometryczny ma co prawda zmienną gęstość punktów, ale ta zmiana jest regularna. Oczywiście nie zawsze istnieje możliwość tak swobodnego doboru punktów (np. gdy dostępny są tylko wybrane instancje dla konkretnych rozmiarów problemu).

Jeśli nie mamy przesłanek dotyczących zakresu x, który powinien dawać najlepszy efekt, to konieczne może być zbadanie dość szerokiego zakresu. Czasami jednak mamy takie przesłanki. Przykładowo popularne wartości prawdopodobieństwa mutacji p_m w algorytmie genetycznym to 0.01



Rysunek 1: Wykres z „błędnym” oznaczeniem osi X

do 0.05. W takim przypadku dla zobrazowania dlaczego wybór wartości spoza tego przedziału jest zły (i czy w ogóle jest zły), można rozszerzyć badanie także poza ten przedział, ale z rzadszym rozmieszczeniem punktów:

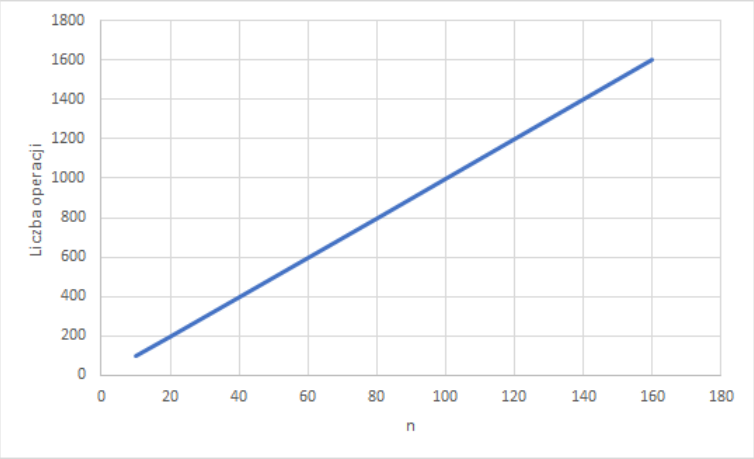
$$p_m \in \{0.001, 0.005, 0.01, 0.02, 0.03, 0.04, 0.05, 0.07, 0.1, 0.15, 0.25, 0.4\}.$$

3 Skale i oznaczenie wykresu

Jednym z częściej spotykanych problemów jest odpowiednie umiejscowienie wartości na osiach oraz dobór skali osi. Jest to o tyle trudne, że rzadko powoduje faktyczny błąd (tzn. dane na wykresie są cały czas poprawne), ale prowadzi do wyciągania błędnych wniosków lub utrudnia rozróżnienie elementów wykresu. Możemy tutaj wyróżnić dwa osobne typy problemów.

Pierwszy problem najczęściej spowodowany jest nieuważnym ustawieniem zawartości etykiet osi X w arkuszu Excela. Za przykład niech posłuży wykres przedstawiony na Rysunku 1. W teorii wykres ten jest poprawny – punkty mają odpowiednie wartości na obu osiach, obrazując zadaną funkcję. Problemem jednak jest sama oś X. Na wykresie punkty na tej osi są od siebie oddalone o taką samą odległość niezależnie od faktycznej różnicy między wartościami punktów tzn. różnica $160 - 80 = 80$ wygląda na taką samą co $20 - 10 = 10$, co jest nieprawdą. Innymi słowy, wielkości które umieszczamy na osi X są w skali interwałowej, zaś wykres wygląda jakby były one w skali nominalnej. Podobny efekt zachodzi, gdy na osi X zamiast liczb oznaczających rozmiar problemu umieszcza się etykiety zastępcze (np. „Instancja A”, „Instancja B” itd.), gubiąc sens wielkości na osi X. W rezultacie próba interpretacji kształtu wykresu prowadzi do mylnego wniosku że przedstawiona została funkcja kwadratowa. Bardziej poprawny sposób przedstawia Rysunek 2. Wartości na osi X są odpowiednio odległe od siebie, co uwidacznia, że na wykresie przedstawiono funkcję liniową. Istotnie, do stworzenia wykresu posłużono się funkcją $f(x) = 10x$.

Drugi problem dotyczy niestosowania skali logarytmicznej tam gdzie jest ona potrzebna. Pojawia się to najczęściej podczas próby porównania czasu wykonywania kilku algorytmów o zupełnie różnej złożoności obliczeniowej. Czasami dochodzi do tego fakt niewielkiej liczby punktów danych. Może się to zdarzyć np. dla algorytmu przeglądu zupełnego, gdzie wyniki dla rozmiaru problemu poniżej $n = 5$ są nieodróżnialne od zera, a powyżej $n = 13$ pomiar zajmuje już bardzo dużo czasu. Zauważmy, że ze względu na dyskretny charakter n, liczba punktów pomiaru nie będzie większa niż 9 (zakładamy tutaj dla uproszczenia, że dla wszystkich algorytmów przyjęto to samo maksymalne n, w praktyce nie jest to zwykle prawda), a wzrost z 5 do 13 jest mniej niż trzykrotny.



Rysunek 2: Wykres z poprawnym oznaczeniem osi X

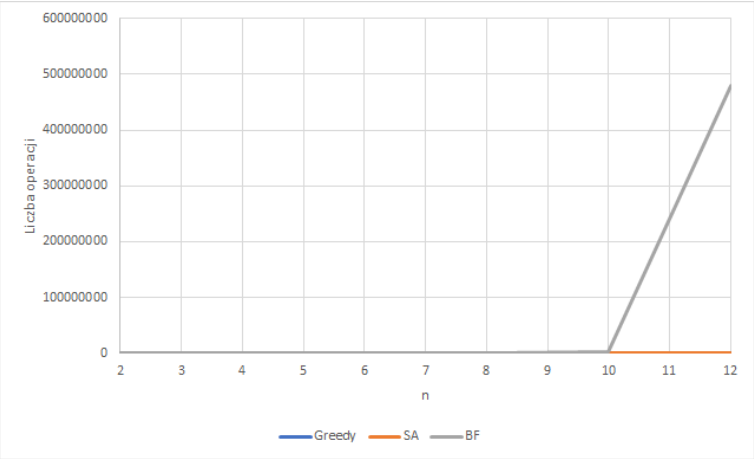
Pokażmy to na przykładzie. Załóżmy że mamy trzy algorytmy. Pierwszym jest algorytm zachłanny (Greedy) o liczbie operacji $15n^2 + 50$ (złożoność kwadratowa), drugim symulowane wyżarzanie (SA), o liczbie operacji $5000n + 200$ (złożoność liniowa) oraz algorytm przeglądu zupełnego o liczbie operacji $n!$. Próba przedstawienia takich złożoności przy tradycyjnej (liniowej) skali osi Y da wynik jak na Rysunku 3. Wykres taki ma co najmniej 3 wady: 1) prawie cała jego powierzchnia jest niewykorzystana, 2) niewidoczny jest kształt linii, 3) niewidoczne są różnice pomiędzy algorytmami, w szczególności pomiędzy SA i Greedy. Wystarczy jednak zmienić skalę osi Y na logarytmiczną, aby usunąć wszystkie powyższe problemy, uzyskując efekt jak na Rysunku 4. Wadą jest trudniejsze uchwycenie złożoności SA i Greedy (linie są podobne), ale można w tym celu skorzystać z osobnego wykresu lub umieścić „w tle" odpowiednie funkcje porównawcze.

Osobną kwestią jest odpowiednie oznaczenie wykresu. W każdym przypadku osie wykresu powinny być podpisane (informować jaka wartość odkładana jest na danej osi). Często też podaje się jednostkę, zwłaszcza w przypadku pomiaru czasu. Wyjątkiem są wartości bezwymiarowe (np. liczba operacji). Jeśli na wykresie przedstawiono wiele serii danych (np. wiele linii) to powinna być umieszczona legenda, informująca czego dotyczy dana seria/linia. Serie/linie powinny też być łatwo rozróżnialne (najczęściej kolorem). Ogólną zasadą też jest możliwie pełne wykorzystanie powierzchni wykresu. Nie zawsze jest to możliwe, ale należy chociaż zadbać, by ręcznie skorygować zakres wartości na obu osiach, gdyż Excel potrafi domyślnie tworzyć wykresy od $x = 0$ i $y = 0$, dla danych gdzie wartość x jest w przedziale od 0.7 do 1, a y od 150 do 250.

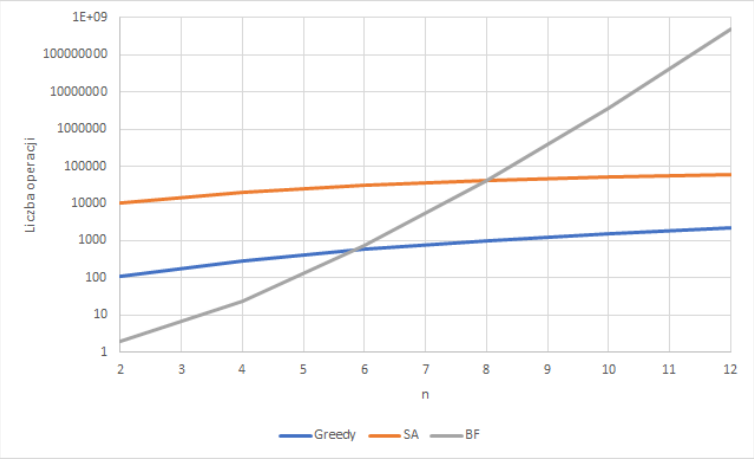
4 Wykorzystanie wielu serii danych

Często istnieje potrzeba zbadania zależności od więcej niż jednej zmiennej. Przykładem może być zbadanie jakości wyników algorytmu symulowanego wyżarzania dla problemu komiwojażera w zależności od rozmiaru problemu (n) oraz rodzaju sąsiedztwa. Problem ten można rozwiązać na kilka sposobów. Można obie zależności zbadać osobno. Można też posłużyć się wykresem 3D. Jednakże najczęściej wybieranym wyjściem jest przedstawienie jednej zmiennej na osi X, a drugiej osobnymi seriami (osobna linia dla każdej wartości zmiennej). Należy wtedy wybrać która zmienna zostanie oznaczona na osi X, a która seriami danych.

Zauważmy że w naszym przykładzie zmienne mają odmienny charakter. Rozmiar problemu n, rozumiany w tym momencie jako liczba miast lub liczba wierzchołków grafu, jest zwykłą liczbą naturalną (skala interwałowa), zaś rodzaj sąsiedztwa jest w stylu „sąsiedztwo typu A", „sąsiedztwo



Rysunek 3: Wykres liczby operacji algorytmów w zależności od n przy liniowej skali osi Y

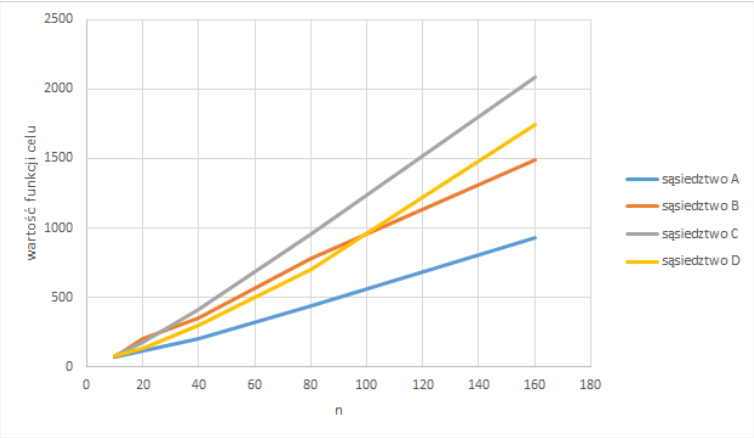


Rysunek 4: Wykres liczby operacji algorytmów w zależności od n przy logarytmicznej skali osi Y

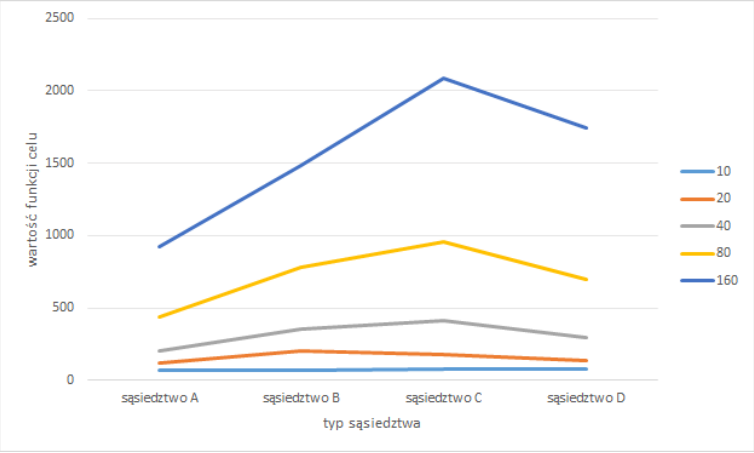
typu B" itd. (skala nominalna). W takim przypadku zmienną w skali interwałowej powinniśmy raczej umieścić na osi X, a zmienną w skali nominalnej osobnymi seriami. Jeśli obie zmienne są zwykłymi liczbami lub obie są w skali nominalnej, to wybór jest mniej oczywisty, niemniej możemy kierować się pewnymi wskazówkami. Ważniejsza zmienna (np. ze względu na cel badań) zwykle pojawia się na osi X. Jest to związane z tym że łatwiej śledzić jej przebieg. Należy też unikać zbyt wielu serii na jednym wykresie. W związku z tym zmienna która ma więcej punktów danych zwykle pojawia się na osi X. Na Rysunkach 5 oraz 6 przedstawiono dwie wersje wykresu dla powyższego zagadnienia.

Oba wykresy pozwalają stwierdzić, że sąsiedztwo A dało najlepszy wynik, że wartość funkcji celu rośnie wraz z rozmiarem problemu oraz że im większy problem tym większe różnice pomiędzy sąsiedztwami. Jednakże tylko pierwsza wersja wykresu pozwala w łatwy sposób wywnioskować że zależność wartości funkcji celu od n jest liniowa.

Drugim zagadnieniem związanym z seriami danych jest łączenie wykresów. Jeśli pojawia się kilka wykresów o tym samych charakterze (np. zależność jakości od α dla danej instancji), to zwykle chcemy porównać te wykresy (np. by stwierdzić lub wykluczyć ogólną zależność). W takim przypadku używanie wielu (np. 5 wykresów) zajmuje bardzo dużo miejsca, zmuszając do częstego przeskakiwania pomiędzy stronami lub plikami. Ponadto pomimo reprezentowania podobnych danych, wykresy mogą



Rysunek 5: Badanie wpływu rozmiaru problemu n i typu sąsiedztwa na jakość wyników, wersja 1



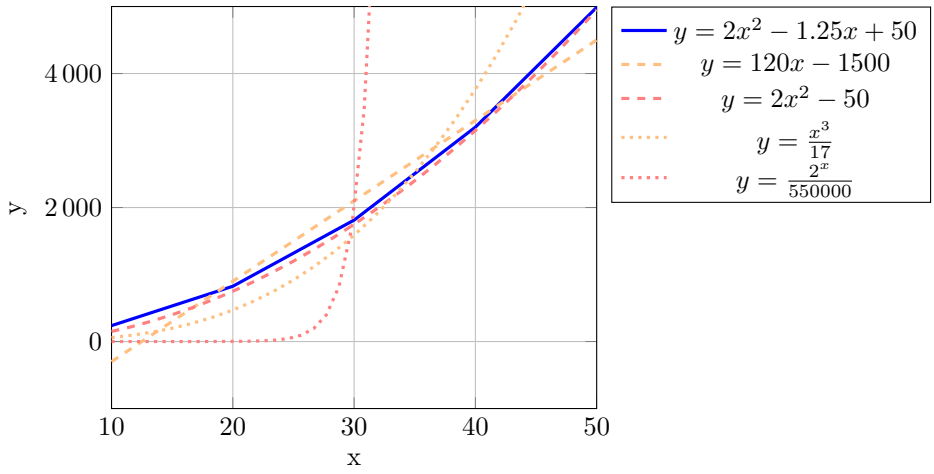
Rysunek 6: Badanie wpływu rozmiaru problemu n i typu sąsiedztwa na jakość wyników, wersja 2

mieć różne zakresy danych na osi Y. Dlatego często lepiej zamiast 5-ciu takich wykresów posłużyć się jednym z 5-cioma seriami danych, co ułatwia analizę. Oczywiście metoda nie jest idealna: czasami różnice pomiędzy poszczególnymi seriami są tak duże lub tak małe, że część serii jest niewidoczna (zbyt blisko osi wykresu lub zakryta przez inną serię). Są to jednak przypadki dosyć rzadkie.

5 Analiza kształtu funkcji

Kolejną kwestią jest wnioskowanie na temat charakteru funkcji (np. jej klasy złożoności) na podstawie danych (linia na wykresie lub tabela). Zagadnienie nie jest proste, gdyż rozróżnienie pomiędzy funkcją kwadratową, sześcienną lub wykładniczą „na oko" utrudniają dodatkowe czynniki rzeczywistej funkcji (np. $2n^2 - 1.25n + 50$) oraz układ wartości na osi X.

Jeśli interesuje nas tylko stwierdzenie czy funkcja jest liniowa czy kwadratowa, to niejednokrotnie do wiarygodnego potwierdzenia (lub bezsprzecznego obalenia) takiej tezy wystarcza prosta analiza danych tabelarycznych. Przykładowo, jeśli chcemy sprawdzić czy funkcja jest liniowa, to wystarczy sprawdzić czy dwukrotny wzrost argumentu (np. z $x = 10$ na $x = 20$) powoduje w przybliżeniu dwukrotny wzrost wartości funkcji. Można to łatwo przenieść na funkcję kwadratową (tutaj dwu-



Rysunek 7: Linie trendu wybranych funkcji porównawczych

krotny wzrost x powinien powodować około czterokrotny wzrost $f(x)$) lub sześcienny (tutaj $f(2x)$ powinno wynosić około $8f(x)$). Trudniej jest dla innych funkcji (np. funkcja wykładnicza) lub gdy nie mamy odpowiedniej wartości x (np. mamy $x = 10$, ale nie mamy $x = 20$, tylko $x = 18$ i $x = 22$). W takich przypadkach można jeszcze zastosować zasadę ogólniejszą. Jeśli naszą funkcją danych jest $g(x)$, a testujemy czy pasuje ona do funkcji $f(x)$, to wystarczy dla odpowiednich par punktów x_1 oraz $x_2 > x_1$ sprawdzić czy zachodzi:

$$\frac{g(x_2)}{g(x_1)} \stackrel{?}{\approx} \frac{f(x_2)}{f(x_1)}. \tag{1}$$

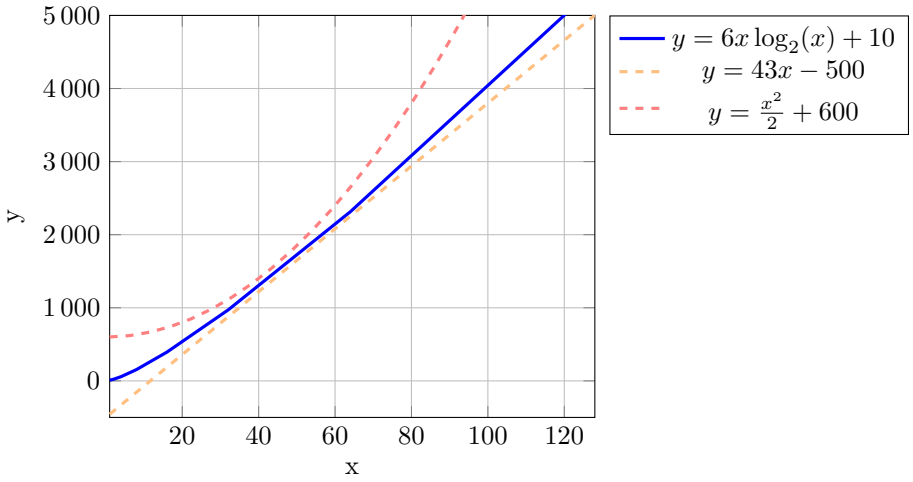
Dla powyższego przypadku (test funkcji kwadratowej, $x_2 = 22$, $x_1 = 10$) sprowadzałoby się to sprawdzenia czy zachodzi:

$$\frac{g(x_2)}{g(x_1)} \stackrel{?}{\approx} \frac{22^2}{10^2} \stackrel{?}{\approx} 4.84. \tag{2}$$

Powyższą metodę można stosować dla wielu funkcji (oczywiście należy sprawdzić to dla więcej niż jednej pary x, zwłaszcza że im x bliższe zeru tym większy wpływ stałych i niepewności związanej z działaniem innych aplikacji komputera). Metoda ta zadziała niezależnie od współczynnika przy najwyższym czynniku (tzn. zarówno dla x^2, $1000x^2$ czy $\frac{1}{1000}x^2$). Niestety nie jest odporna na stałą (tzn. x^2, $x^2 - 1000$, $x^2 + 1000$). Stała powoduje że iloraz (np. 4 dla funkcji kwadratowej) będzie inny niż spodziewany. Efekt ten co prawda staje się pomijalny dla odpowiednio wysokich wartości x, ale nie zawsze możliwe jest ich uzyskanie w pomiarach.

Z kolei metodą dla wykresów jest porównanie wykresu funkcji do pomocniczej funkcji (trendu). W przeciwieństwie do wykresu pomiarowego (który jest rysowany ciągłą linią łamaną na podstawie punktów danych), wykres trendu zwykle rysowany jest linią wygładzoną (krzywą) kreskowaną lub kropkowaną. Przykład próby określenia złożoności dla funkcji $2x^2 - 1.25x + 50$ został przedstawiony na Rysunku 7. Przedstawione tam zostały 4 linie trendu (liniowa, kwadratowe, sześcienna i wykładnicza). W praktyce testuje się w danym momencie tylko jedną klasę (np. kwadratową), a w przypadku braku dopasowania przechodzi się do kolejnej klasy. Formalnie celem jest takie ustawienie parametrów (współczynniki, stała) funkcji porównawczej by odległość obu funkcji (najczęściej w sensie błędu kwadratowego) była minimalna. W praktyce chcemy by obie funkcje miały zbliżone wartości w centrum wykresu i możliwie podobny kształt. Na podanym wykresie jedynym wyjściem jest funkcja kwadratowa, ewentualnie sześcienna.

Zbliżoną metodą jest określenie funkcji ograniczających zadaną z dołu i z góry. Np. na Rysunku 8 przedstawiono funkcję liniowo-logarytmiczna oraz 2 funkcje pomocnicze: liniową i kwadratową.



Rysunek 8: Linie trendu wybranych funkcji porównawczych

Różnica pomiędzy funkcją liniową i liniowo-logarytmiczną jest trudniejsza do zauważenia, co wynika częściowo z zakresu osi X (różnica była lepiej widoczna gdyby X obejmowało kilka rzędów wielkości).

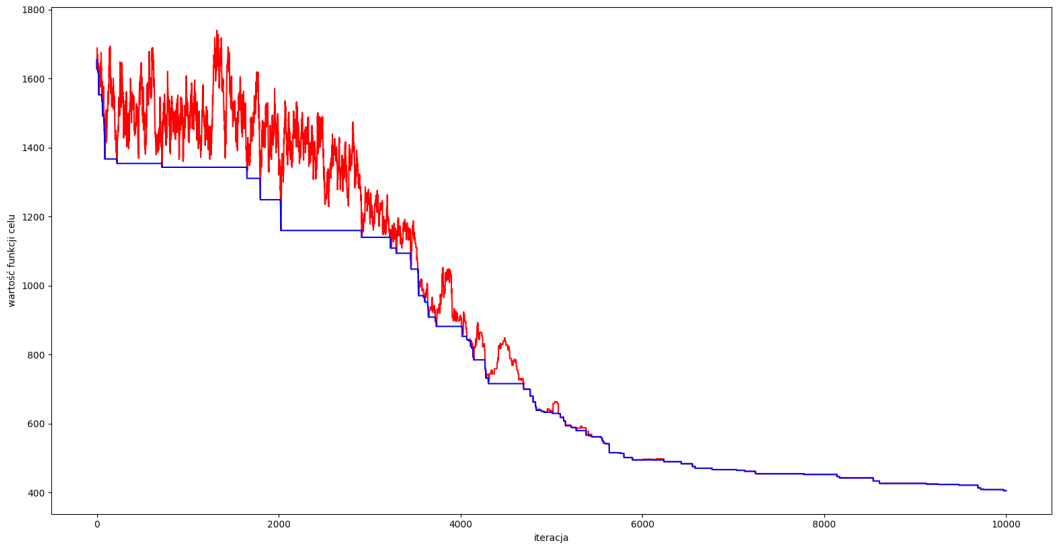
6 Porównywanie wyników

Jeśli chcemy rzetelnie porównywać wyniki działania algorytmów (lub przynajmniej wiedzieć na ile uprawnione są wyciąganie wnioski), należy zwrócić uwagę na szereg okoliczności opisanych poniżej.

Zwykle chcemy oceniać albo jakość algorytmów (wartość zwracanej funkcji celu), albo ich czas działania. Załóżmy, że chcemy porównać jakość. W takim przypadku najlepiej jeśli uruchomione algorytmy mają możliwie zbliżony czas działania. Analogicznie, jeśli chcemy porównać czas działania, to algorytmy powinny dawać wyniki o zbliżonej jakości. Niestety, nie zawsze jest to łatwe do uzyskania (szczególnie dla tego drugiego scenariusza). Dodatkowo, jeśli badane algorytmy mają różną złożoność (np. jeden jest liniowy względem rozmiaru problemu n a drugi kwadratowy), to dobranie identycznego czasu jest bardzo trudne przy zmiennym n. Jeśli nie uda się ustawić „wspólnego mianownika" to może dojść do sytuacji że jeden z algorytmów będzie szybszy, ale będzie dawał gorsze wyniki, utrudniając stwierdzenie który algorytm jest lepszy.

Zauważmy też ze dla wielu algorytmów zależność jakość-czas jest specyficzna. Względnie duże (np. kilkukrotne) zwiększenie czasu działania, powoduje ograniczony (niewielki, niekiedy żaden) wzrost jakości wyników. Powyższe zjawisko często uwidacznia się przy niewłaściwym doborze parametrów algorytmów, powodując jego przedwczesną zbieżność. Taką sytuację dość łatwo zdiagnozować, sprawdzając kiedy następuje ostatnia poprawa. Jeśli na 10 000 wykonanych iteracji poprawa występuje tylko do iteracji nr 200 (czyli 98% czasu działania algorytmu się marnuje), to należy wprowadzić zmiany w algorytmie, a w ostateczności skrócić czas działania.

Przykładowo, na Rysunku 9 przedstawiono przebieg działania pojedynczego uruchomienia algorytmu symulowanego wyżarzania z limitem 10 000 iteracji dla pewnej instancji problemu komiwojażera. Niebieska linia przedstawia najlepsze znane rozwiązanie (jest więc funkcją nierosnącą), zaś czerwona oznacza rozwiązanie aktualne. Widać że częstość występowania popraw jest różna w zależności od fazy algorytmu, ale występują one nawet tuż przed wykonanej wszystkich iteracji. Tego typu wykresy dostarczają sporo informacji o działaniu algorytmu, włącznie z całkowitą poprawą (koniec algorytmu względem początku): tutaj wartość funkcji celu poprawiła się około 4-krotnie. Zasadniczo wykonuje się je jednak dla pojedynczego (przykładowego) uruchomienia algorytmu lub dla diagnostyki. Ciężko więc na ich podstawie wyciągać ogólne wnioski o skuteczności algorytmu.



Rysunek 9: Aktualne i najlepsze znane rozwiązanie w przebiegu algorytmu SA

Kolejnym czynnikiem istotnym przy ocenie pracy algorytmu jest stabilność wyników. Innymi słowy, czy algorytm uruchomiony drugi raz dla tych samych danych wejściowych (ta sama instancja problemu) zachowa się tak samo? W tym miejscu warto zdefiniować pojęcie trajektorii. Podstawowa definicja najlepiej sprawdza się dla algorytmów typu symulowane wyżarzanie, które mają w danej chwili jedno rozwiązanie (inaczej jest w przypadku algorytmów populacyjnych gdzie jednocześnie istnieje wiele rozwiązań). Trajektoria jest wtedy po prostu ciągiem rozwiązań, które algorytm odwiedza w trakcie swojego działania (czego śladem jest np. czerwona linia na Rysunku 9. Z niej zresztą wynika niebieska trajektoria najlepszego znanego rozwiązania).

W przypadku algorytmów deterministycznych trajektoria jest zawsze ta sama (jeśli nie zmieniliśmy parametrów algorytmu ani danych wejściowych), co oznacza że jakość algorytmu jest stabilna (zwrócona wartość funkcji celu nie zmienia się po ponownym wykonaniu algorytmu). Podobnie jest w kwestii czasu działania: jedyne różnice wynikają z czynników zewnętrznych (inne aplikacje uruchomione na komputerze itd.). Takie czynniki mogą mieć znaczący wpływ na pomiar czasu jeśli czas działania programu jest bardzo niski lub inne aplikacje aktywnie wykorzystują czas procesora. W innym przypadku można je zignorować i założyć że czas działania algorytmu jest stabilny.

Inaczej sprawa wygląda w przypadku algorytmów probabilistycznych, czyli takich które podczas działania korzystają z generatora liczb (psuedo)losowych. Jeśli przy dwukrotnym uruchomieniu wygenerowane zostaną te same liczby pseudolosowe w tej samej kolejności (czyli generator ma to samo tzw. ziarno) to trajektoria będzie ta sama. Zmiana ziarna może prowadzić do zmiany trajektorii, a w konsekwencji do zmiany wyniku. Dodatkowo, jeśli warunek stopu jest nietypowy (np. przekroczony limit liczby iteracji od ostatniej poprawy), to zmianie może ulec też czas działania algorytmu. Z jednej strony powyższe zjawisko pozwala algorytmom losowym na łatwiejsze unikanie pewnych pułapek, w które wpadają algorytmy deterministyczne. Z drugiej strony takie zachowanie nie jest powtarzalne i utrudnia wyciąganie wniosków podczas badań.

Pierwszym, mniej skutecznym rozwiązaniem tej kwestii, jest wykazanie, że niestabilność algorytmu jest niska. W tym celu można zrobić tzw. badania wstępne, uruchamiając algorytm wielokrotnie (co najmniej 5 razy) dla tych samych danych i parametrów, ale o innym ziarnie i sprawdzaniu np. wariancji czy odchylenia standardowego wyniku (lub czasu działania). Jeśli wykonamy to dla kilku zestawów danych wejściowych/parametrów i wariancja będzie cały czas niska, to można założyć że algorytm jest stabilny i postępować dalej jak dla algorytmów deterministycznych.

Najczęściej jednak niestabilność jest znaczna. W takim przypadku można jeszcze próbować ustabilizować algorytm np. zwiększając liczbę iteracji. To jednak nie zawsze pomaga, a wydłuża niepotrzebnie czas działania. Najprostszym wyjściem, które zadziała praktycznie w każdych warunkach, lecz wymaga więcej czasu na badania, jest uruchamianie algorytmu wielokrotnie, dla różnych ziaren generatora psuedolosowego. Liczba powtórzeń nie powinna być mniejsza niż 10 (a na pewno nie mniejsza niż 5). Przykładowo załóżmy, że dla danej instancji problemu komiwojażera i danych parametrów algorytmu symulowanego wyżarzania uruchomiono go 10-krotnie (z różnym ziarnem) otrzymując następujące wyniki:

$$(110, 108, 121, 98, 105, 120, 115, 102, 118, 105).$$

Interpretacji tych wyników nie jest do końca oczywista. Teoretycznie najlepszym wynikiem jest 98 (problem komiwojażera jest problemem minimalizacji), jednakże nie mamy pewności uzyskania tego wyniku. Dopiero po pewnej liczbie prób nabędziemy odpowiednio poziom pewności uzyskania tak dobrego wyniku. W uproszczeniu oznacza to, że możemy przyjąć jako wynik wartość 98, ale przy założeniu, że algorytm był uruchamiany 10 razy, czyli wykonywał się de facto 10 razy dłużej niż analogiczny algorytm deterministyczny.

Możemy zastosować też myślenie odwrotne. Najgorszą wartością uzyskaną w tej próbie jest 121. Możemy więc powiedzieć, że wynikiem jest 121, gdyż przy pojedynczym uruchomieniu nie dostaniemy raczej (ponownie w uproszczeniu) gorszej wartości. Czas wykonania zostaje bez zmian.

Powyższe dwie opcje odpowiadają w przybliżeniu analizie najgorszego i najlepszego przypadku. Kolejną, bardzo często wykorzystywaną opcją, jest analiza przypadku średniego. W tym przypadku średnia z wyniku to 110.2 i stanowi przybliżenie (estymator) wartości oczekiwanej. Wartość tą można uzupełnić odchyleniem standardowym.

Uśrednianie wyników wielokrotnego uruchomienia algorytmu dla tych samych danych wejściowych i tych samych parametrów algorytmów jest trywialne (zwykła średnia arytmetyczna). Nawet zmiana parametrów algorytmu nie powoduje tutaj problemów. Inaczej może być jednak jeśli zmieniły się dane wejściowe (instancja), w szczególności jej rozmiar. Porównywanie ze sobą wyników dla instancji komiwojażera dla 10 miast oraz 100 miast nie powinno być dokonywane wprost, gdy typowo uzyskiwane wartości funkcji celu będą różne (w tym przypadku 10-krotnie). Wtedy wyniki dla instancji rozmiaru 100 mają dużo większy wpływ na wynik. Ponadto, nawet dwie instancje o tym samym rozmiarze mogą mieć skrajnie różne wartości funkcji celu (wystarczy wziąć ten sam graf dla problemu komiwojażera i pomnożyć wagi wszystkich krawędzi przez np. 100).

Próba uśredniania wyników pochodzących z różnych instancji przy użyciu zwykłej średniej niesie ryzyko, że faktyczny charakter zjawiska zostanie zniekształcony przez różne „skale" poszczególnych instancji. Jednym z rozwiązań tego problemu jest normalizacja danych. Innymi słowy, wszystkie wartości funkcji celu uzyskane na podstawie danej instancji I powinny być podzielone przez pewien czynnik f_I. Jako f_I można wybrać najlepszy wynik spośród wszystkich uzyskanych dla instancji f_I.

Rozważmy to na przykładzie. Załóżmy że mamy dwie instancje oznaczone A, B. Dla każdej instancji badano dwie wersje algorytmu różniące się parametrami, nazwijmy je wariantem α i β. Ponadto ze względu na losowy charakter algorytmu każde uruchomienie było wielokrotne. Dla zwięzłości załóżmy że były to tylko 3 powtórzenia. Powiedzmy, że uzyskano następujące wyniki:

$$A_\alpha = (105, 91, 102), \qquad A_\beta = (103, 115, 125),$$
$$B_\alpha = (8231, 7985, 8315), \qquad B_\beta = (7923, 8034, 8102),$$

Naszym celem jest zbadanie, który zestaw parametrów, α czy β, daje lepsze wyniki. Pobieżny wgląd w dane pokazuje że beta jest gorsza dla instancji A, ale lepsza dla instancji B. Chcemy ten wniosek uogólnić, więc liczymy średnią. Każdy zestaw ma przypisane 6 wartości (2 instancje razy 3 powtórzenia). Zastosowanie średniej wprost da nam wynik 4138.17 dla α oraz 4067.00 dla β, prowadząc do wniosku że ten drugi parametr jest nieznacznie lepszy. Jest to jednak iluzja spowodowana większymi bezwzględnymi wartościami instancji B.

Spróbujmy teraz podejścia znormalizowanego. Najpierw dla każdej instancji określamy najlepszy dla niej wynik. Jest to 91 dla instancji A, oraz 7923 dla instancji B. Następnie normalizujemy wyniki w każdej instancji, dzieląc je przez odpowiednią z tych dwóch wartości. Wedy wartość 1.0 oznacza najlepszy uzyskany wynik dla danej instancji, a przykładowo 2.0 oznacza wynik 2 razy gorszy.

Po normalizacji (zaokrąglenie do dwóch miejsc po przecinku) dane wyglądają następująco.

$$A_\alpha = (1.15, 1.00, 1.12), \qquad A_\beta = (1.13, 1.26, 1.37),$$
$$B_\alpha = (1.04, 1.01, 1.05), \qquad B_\beta = (1.00, 1.01, 1.02).$$

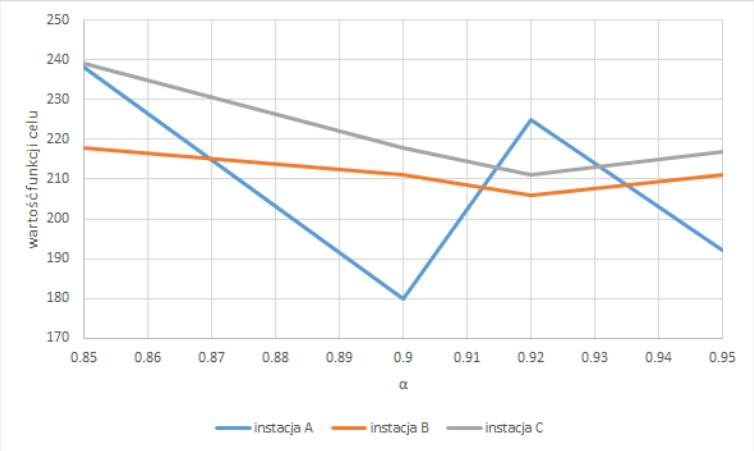
Teraz obliczamy średnią z wartości znormalizowanych otrzymując 1.06 dla α oraz 1.13 dla β. Wynik jest więc odmienny, a różnica bardziej znacząca.

Po przeczytaniu powyższego przykładu może powstać pytanie: dlaczego w ogóle testować różne instancje problemu, skoro powoduje to problemy z ujednoliceniem i interpretacją wyników? Otóż zwykle chcemy wykazać ogólną cechę algorytmu (lub jego wariantu), która jest spełniona dla całego problemu, niezależnie od danych wejściowych. Dlatego testowanie tylko jednej instancji dla jednego rozmiaru problemu i uruchomionej jeden raz (w przypadku algorytmów losowych) może zaburzyć wyniki i doprowadzić do wyciągania błędnych wniosków.

Pokażmy to na przykładzie. Załóżmy, że chcemy sprawdzić jaki wpływ na jakość algorytmu symulowanego wyżarzania dla problemu komiwojażera ma współczynnik chłodzenia α. Warunkiem stopu było przekroczenie zadanej liczby iteracji równej 10 000. Powiedzmy, że wybrano następujące wartości α do zbadania:

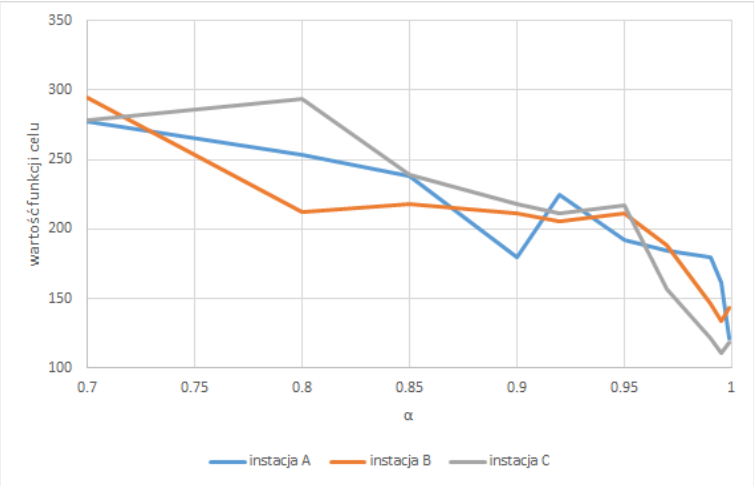
$$\alpha \in \{0.7, 0.8, 0.85, 0.9, 0.92, 0.95, 0.97, 0.99, 0.995, 0.999\}.$$

Załóżmy teraz że uruchomiono ten algorytm dla 3 różnych instancji, ale o tym samym rozmiarze problemu (30 miast), każdą uruchamiając tylko raz. Co więcej chwilowo załóżmy, że badanie przeprowadzono dla węższego zakresu $\alpha \in \{0.85, 0.9, 0.92, 0.95\}$. Wynik przedstawiono na Rysunku 10. Wykres ten nie pozwala wyciągnąć żadnego wniosku, poza tym że wpływ α na jakość jest losowy lub nieokreślony. Jeśli poszerzymy zakres α jak pokazano na Rysunku 11, to sytuacja wygląda lepiej. Ogólna zależność malejąca jest widoczna, ale instancje zachowują się każda inaczej. Poza tym, linie wykresu są dalekie od gładkich, tworząc wyraźną linię łamaną.



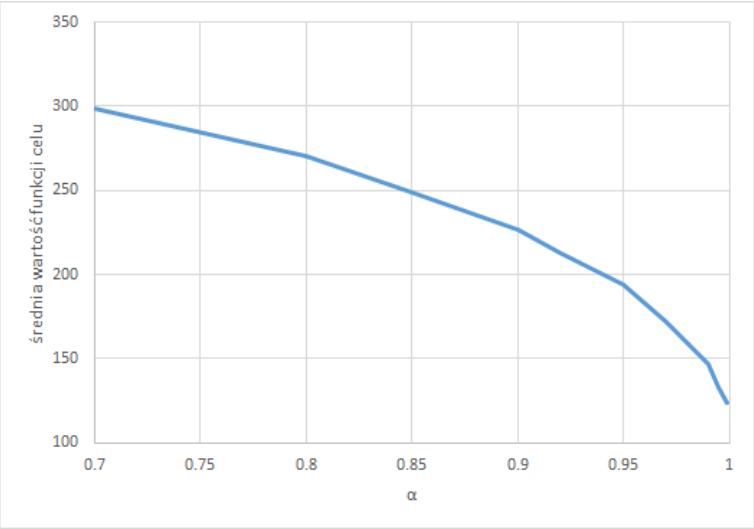
Rysunek 10: Badanie wpływu α na jakość wyników: 3 osobne instancje, jeden rozmiar problemu, wąski zakres α

Z kolei na Rysunku 12 przedstawiono wynik w postaci jednej linii. W tym przypadku wykorzystano 5 różnych rozmiarów problemu (od 10 do 50 miast), po 10 różnych instancji na rozmiar i po



Rysunek 11: Badanie wpływu α na jakość wyników: 3 osobne instancje, jeden rozmiar problemu, szeroki zakres α

10 powtórzeń na instancję. W tym przypadku wynikiem jest zadziwiająco „gładki" wykres, z którego wnioski są oczywiste: wyższe (bliższe 1) wartości α pozwalają uzyskać lepszą wartość funkcji celu, zaś najlepsze wyniki uzyskano dla $\alpha = 0.999$. Co więcej spadek wartości funkcji celu względem $\alpha = 0.7$ był około 2.5-krotny. W tym przypadku również można zastosować normalizację, ze względu na stosowanie różnych instancji i rozmiarów problemu, ale wykres po takim zabiegu był w zasadzie identyczny (pomijając zmianę wartości na osi Y). Wadą takiego podejścia jest znaczna liczba obliczeń: w tym przypadku algorytm był uruchamiany $5 \times 10 \times 10 \times 10 = 5000$ razy. Można oczywiście zmniejszyć rozmiar badań (np. $3 \times 5 \times 5 \times 10 = 750$ uruchomień), ale należy pamiętać, że przy małym rozmiarze próby wyniki mogą zostać zniekształcone.



Rysunek 12: Badanie wpływu α na jakość wyników: 5 rozmiarów problemu $\times$ 10 instancji $\times$ 10 powtórzeń, szeroki zakres α

Na zakończenie dyskusji o trajektoriach i niestabilności algorytmów warto zwrócić uwagę na

jeden fakt. Czasami chcemy sprawdzić jak czas działania lub liczba iteracji algorytmu wpływa na wyniki. W tym celu mierzymy wynik dla kilku różnych liczb iteracji (np. 100, 200, 400, 800 iteracji). Zauważmy, że można to zrobić na dwa sposoby:

1. Uruchomić algorytm 4 razy, za każdym razem zmieniając odpowiedni parametr.
2. Uruchomić algorytm raz dla 800 iteracji i zapisać wartości uzyskane dla pozostałych liczb iteracji (100, 200, 400) w trakcie tego samego przebiegu algorytmu.

W obu przypadkach raportujemy najlepsze znane rozwiązanie.

Jeśli stosujemy to samo ziarno generatora lub algorytm jest deterministyczny to oba podejścia są równoważne. Jeśli jednak stosujemy różne ziarna (i co gorsza nie stosujemy powtórzeń), to czasami podejście pierwsze może dać pozornie niemożliwy wynik: rozwiązanie dla 400 iteracji może być lepsze niż dla 800. Może się tak stać jeśli dla 400 i 800 iteracji wykonane zostały inne trajektorie.