2-aproksymacyjny algorytm dla problemu $R||C_{\max}$

Maksymilian Neumann

4 czerwca 2025

1 Wprowadzenie

Celem zadania jest zaimplementowanie algorytmu 2-aproksymacyjnego dla problemu szeregowania zadań na niezależnych maszynach (model $R||C_{\max}$), gdzie każde zadanie może mieć różny czas wykonania na każdej z maszyn. Celem jest minimalizacja czasu zakończenia ostatniego zadania (makespanu).

Algorytm bazuje na programowaniu liniowym oraz procedurze przedstawionej w książce *Vazirani, Approximation Algorithms* [1], rozdział 17, algorytm 17.5. Składa się z kilku kroków:

- określenia zakresu wartości T,
- wyszukania minimalnego T^*, dla którego relaksacja LP(T) jest wykonalna,
- pozyskania ekstremalnego punktu rozwiązania LP(T^*),
- zaokrąglenia rozwiązania ułamkowego poprzez dopasowanie doskonałe w grafie dwudzielnym,
- przypisania zadań do maszyn i wyliczenia makespanu.

2 Model matematyczny i algorytm

Rozważamy n zadań oraz m maszyn. Dla każdego zadania i oraz maszyny j znamy czas wykonania $p_{i,j} \geq 0$ (jeśli $p_{i,j} = 0$, oznacza to, że zadania nie można przypisać do maszyny j).

2.1 Relaksacja programowania liniowego LP(T)

Dla ustalonej wartości $T > 0$, sprawdzamy wykonalność następującego programu liniowego:

$$
\begin{aligned}
\text{zmienne: } & x_{i,j} \in [0,1] \quad \forall i \in \{1,\dots,n\},\ j \in \{1,\dots,m\} \\
\text{s.t. } & \sum_{j=1}^{m} x_{i,j} = 1 \quad \forall i \quad \text{(każde zadanie musi być przypisane)} \\
& \sum_{i=1}^{n} p_{i,j} \cdot x_{i,j} \leq T \quad \forall j \quad \text{(obciążenie maszyn nie przekracza } T\text{)} \\
& x_{i,j} = 0 \quad \text{jeśli } p_{i,j} > T
\end{aligned}
$$

2.2 Zakres wartości LP(T)

Algorytm najpierw wyznacza zakres wartości T, dla którego szukamy rozwiązania. W tym celu przypisujemy każde zadanie do maszyny z minimalnym czasem przetwarzania. Definiujemy:

$$\alpha = \max_{j \in \{1,\ldots,m\}} \sum_{i:\min_k p_{i,k}=p_{i,j}} p_{i,j}$$

Następnie przeszukujemy binarnie przedział $[\lceil \alpha/m \rceil, \lceil \alpha \rceil]$, szukając najmniejszego T^*, dla którego LP(T) jest wykonalne.

2.3 Zaokrąglanie rozwiązania

Po znalezieniu ekstremalnego rozwiązania $x^*_{i,j}$, wykonujemy procedurę zaokrąglania, składającą się z następujących etapów:

(a) Wszystkie zadania, dla których istnieje maszyna j z $x_{i,j} \geq 1 - \varepsilon$, przypisujemy deterministycznie do tej maszyny.

(b) Dla pozostałych zadań konstruujemy graf dwudzielny H, w którym wierzchołki odpowiadają zadaniom i maszynom, a krawędzie występują między i i j, jeśli $x_{i,j} > \varepsilon$.

(c) W grafie H wykonujemy procedurę *leaf stripping* — iteracyjnie usuwamy i przypisujemy pary (zadanie, maszyna) z maszynami o stopniu 1.

(d) Następnie dla pozostałych cykli w grafie wykonujemy dopasowanie doskonałe (każde zadanie przypisujemy do jednej z sąsiadujących maszyn w cyklu).

Efektem tej procedury jest przypisanie każdego zadania do dokładnie jednej maszyny. Następnie obliczane są obciążenia maszyn i wyznaczany jest makespan $C_{\max}$ jako maksymalne obciążenie.

3 Implementacja

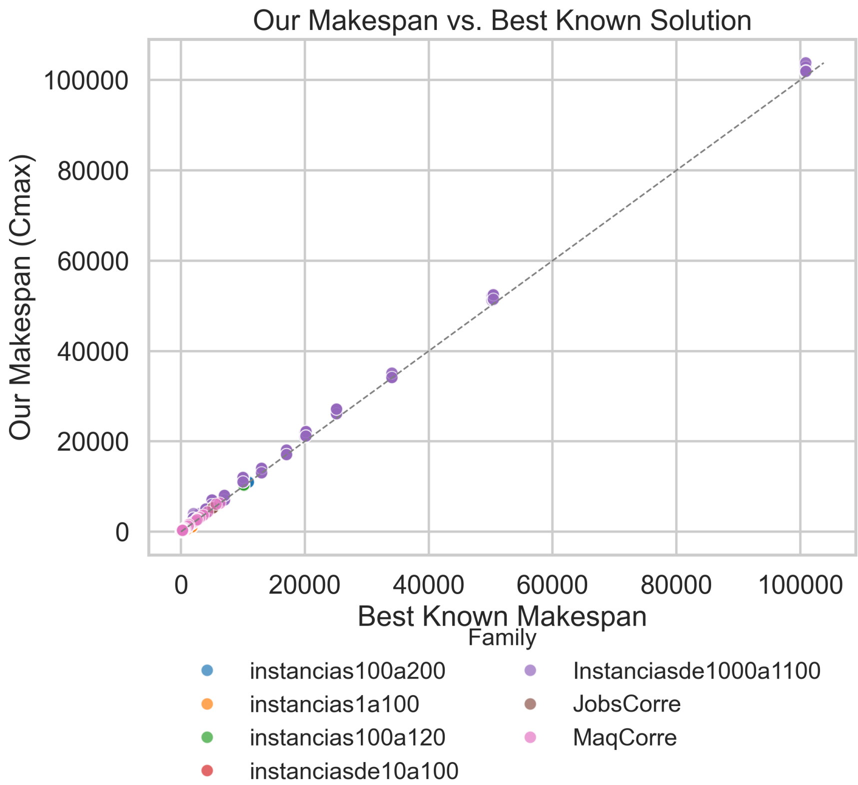
Algorytm został zaimplementowany w języku `Julia` z wykorzystaniem pakietu `JuMP` i solvera `HiGHS`. Cała procedura obejmuje:

- funkcję do wczytywania danych z plików RCmax,
- funkcję testującą wykonalność LP(T),
- binarne przeszukiwanie zakresu T w celu znalezienia T^*,
- zaokrąglanie rozwiązania ułamkowego poprzez konstrukcję grafu i dopasowanie,
- walidację rozwiązania końcowego oraz eksport wyników do pliku `RCmax_summary.csv`.

4 Wyniki

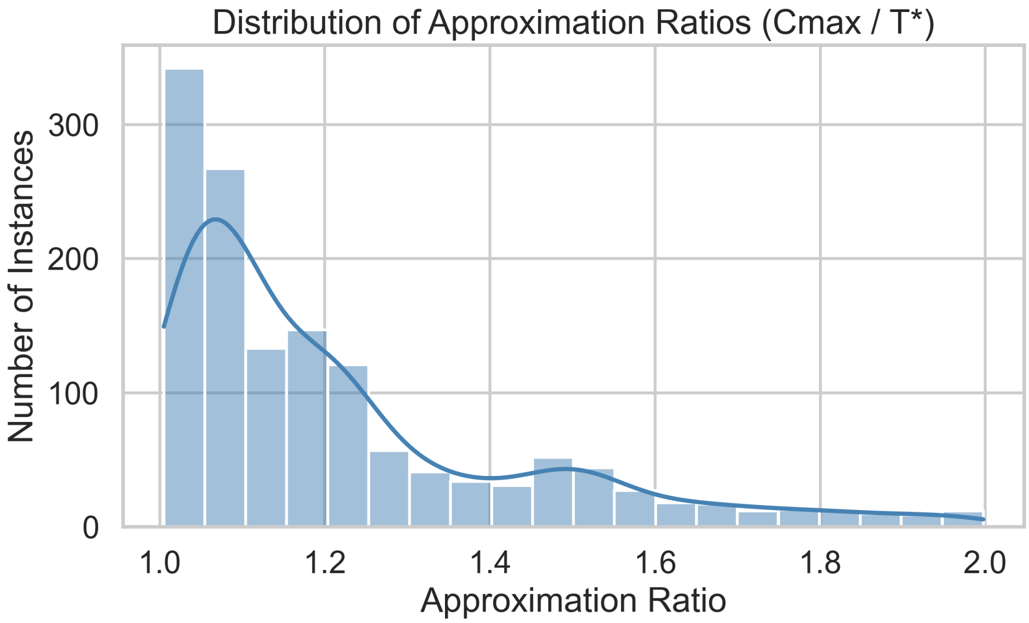
Testy przeprowadzono na zbiorze instancji z RCmax. Łączny czas wykonania obliczeń wyniósł 1 godz. 39 minut. Poniżej zaprezentowano wybrane wykresy i statystyki ilustrujące skuteczność podejścia.

Porównanie makespanów



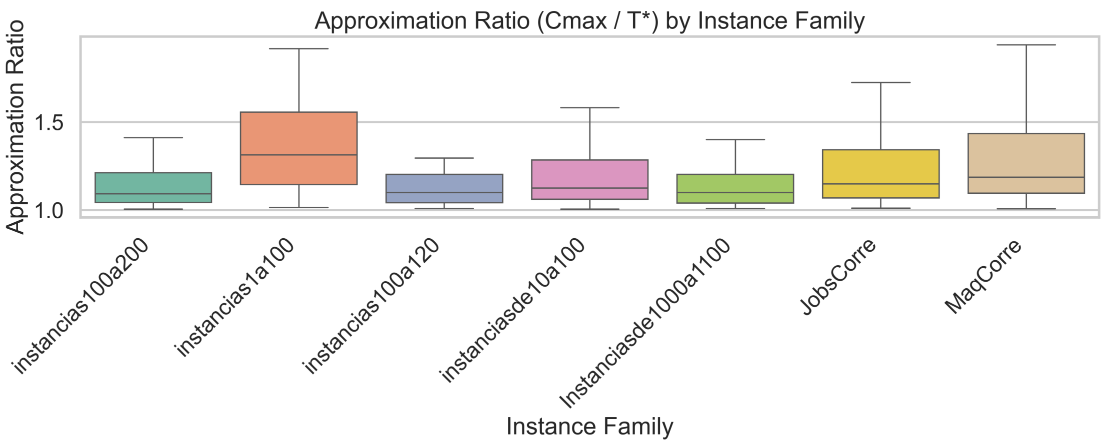
Na powyższym wykresie przedstawiono porównanie wartości $C_{\max}$ uzyskanych przez nasz algorytm względem najlepszych znanych wartości. Idealna zgodność odpowiada linii przekątnej. Większość punktów znajduje się blisko niej, co świadczy o wysokiej jakości aproksymacji.

Rozkład współczynnika aproksymacji



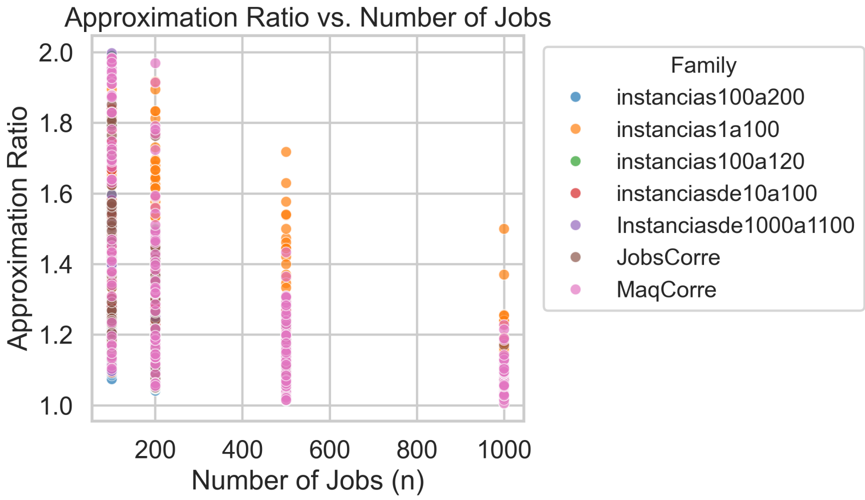
Histogram przedstawia rozkład wartości stosunku $C_{\max}/T^*$. Większość przypadków mieści się poniżej 1.4, co potwierdza praktyczne działanie algorytmu lepsze niż gwarantowane 2.

Wariacje współczynnika względem rodziny instancji



Zaobserwowano, że niektóre rodziny instancji (np. `instancias1a100`) wykazują większą rozpiętość wyników niż inne, co może być związane z ich specyfiką strukturalną.

Zależność jakości od liczby zadań



Na powyższym wykresie nie widać wyraźnego trendu pogarszania się jakości wraz ze wzrostem liczby zadań, co może sugerować skalowalność podejścia.

Statystyki ogólne

- Średni współczynnik aproksymacji: 1.218
- Mediana współczynnika aproksymacji: 1.137
- Maksymalna zaobserwowana wartość: 1.998
- Minimalna różnica względem CPLEX: −837
- Maksymalna różnica względem CPLEX: +2915

Najlepsze i najgorsze przypadki (różnica względem wyników CPLEX)

Najgorsze (Cmax – Best):

Rodzina	Plik	Best	$C_{\max}$	Różnica
Instanciasde1000a1100	1017.txt	100867	103782	+2915
Instanciasde1000a1100	1013.txt	100922	103759	+2837
Instanciasde1000a1100	517.txt	50420	52517	+2097
Instanciasde1000a1100	213.txt	20177	22225	+2048
Instanciasde1000a1100	115.txt	10105	12141	+2036

Najlepsze (Cmax – Best):

Rodzina	Plik	Best	$C_{\max}$	Różnica
instancias1a100	1015.txt	1814	977	−837
instancias1a100	1017.txt	1804	985	−819
instancias1a100	1012.txt	1754	960	−794
instancias1a100	1014.txt	1761	976	−785
instancias1a100	1016.txt	1765	992	−773

References

[1] Vijay V. Vazirani. *Approximation Algorithms*. Springer, 2001. ISBN 978-3-540-65367-7.