Raport z analiz czasow

Jakub Skrajny

January 28, 2020

1 Na poczatek

W kazdym podzadaniu korzystam z rozglaszania logarytmicznego, ktore dzieli przedzial na dokladnie numberOfShamans elementow. Oczywicie dla obliczenia T_∞ nie ma to znaczenia, natomiast wpywa to na czasy testow, przy liczbie szamanow nie bedacej potega dwojki.
We wszystkich testach o numerach 4 oraz 5, najwieksza roznice widac po dodaniu drugiego szamana. Skraca to czas testu o okolo polowe. Dodanie kolejnych szamanow nieznacznie poprawia czas dzialania.

2 botomlessBagTest

Dynamiczny algorytm sekwencyjny dziala w czasie O(S * n), gdzie S - pojemnosc plecaka, n - ilosc roznych jaj.
Algorytm wspolbiezny:
$T_1 = O(S * n)$
$T_\infty = O(\log S * n)$,
Parallelism: $T_1/T_\infty = O(S/\log S)$

3 sandArrangementTest

Algorytm sekwencyjny dziala w czasie O ($n\log n$), gdzie n - ilosc ziarenek piasku.
Algorytm wspolbiezny:
$T_1 = O(n \log n)$
$T_\infty = O(n)$, bo
$f(s) = O(s)$ - merge w sortGrains, gdzie r-f=s
$g(s) = O(s^{\log_2 1}) = O(1)$ - na dwie czescie, dzialamy w jednej(ze wzoru)
Parallelism: $T_1/T_\infty = O(\log n)$

4 crystalSelectionTest

Algorytm sekwencyjny dziala w czasie O(n), gdzie n - ilosc krysztalow.
Algorytm wspolbiezny:
$T_1 = O(n)$
$T_\infty = O(\log n)$
Parallelism: $T_1/T_\infty = O(n/\log n)$