

Profiling

Michal Havlíček, Tomáš Sitarčík, Jan Šemora

profiling.Profiling.main(String[])	20,701	100 %
Profiling.java:19 profiling.Profiling.calculate(ArrayList)	17,658	85 %
Profiling.java:34 profiling.Profiling.average(ArrayList)	5,752	28 %
Profiling.java:56 math.Arithmetic.div(double, double)	1,525	7 %
Profiling.java:41 math.Arithmetic.div(double, double)	1,544	7 %
Profiling.java:42 math.Arithmetic.sqrt(double)	1,530	7 %

Obrázek 1: Výsledky profileru pro 10 vstupních hodnot

profiling.Profiling.main(String[])	28,753	100 %
Profiling.java:19 profiling.Profiling.calculate(ArrayList)	27,400	95 %
Profiling.java:34 profiling.Profiling.average(ArrayList)	13,088	46 %
Profiling.java:56 math.Arithmetic.div(double, double)	688	2 %
Profiling.java:41 math.Arithmetic.div(double, double)	697	2 %
Profiling.java:42 math.Arithmetic.sqrt(double)	683	2 %

Obrázek 2: Výsledky profileru pro 1000 vstupních hodnot

profiling.Profiling.main(String[])	24,544	100 %
Profiling.java:19 profiling.Profiling.calculate(ArrayList)	24,538	99 %
Profiling.java:34 profiling.Profiling.average(ArrayList)	9,460	39 %
Profiling.java:56 math.Arithmetic.div(double, double)	3	0 %
Profiling.java:42 math.Arithmetic.sqrt(double)	2	0 %
Profiling.java:41 math.Arithmetic.div(double, double)	0.5	0 %

Obrázek 3: Výsledky profileru pro 1000000 vstupních hodnot

S přibývajícímí hodnotami se snižuje relativní doba vykonání matematických operací, zatímco se zvyšuje relativní doba samotné iterace. Tato skutečnost je nejspíš způsobena pomalým čtením z paměti. Podstatnou část doby běhu programu zabírá funkce `average`, která počítá aritmetický průměr hodnot. Jako optimalizaci by bylo možné hodnoty procházet pouze jednou, a průměr počítat průběžně, což by zamezilo druhému průchodu skrz data.