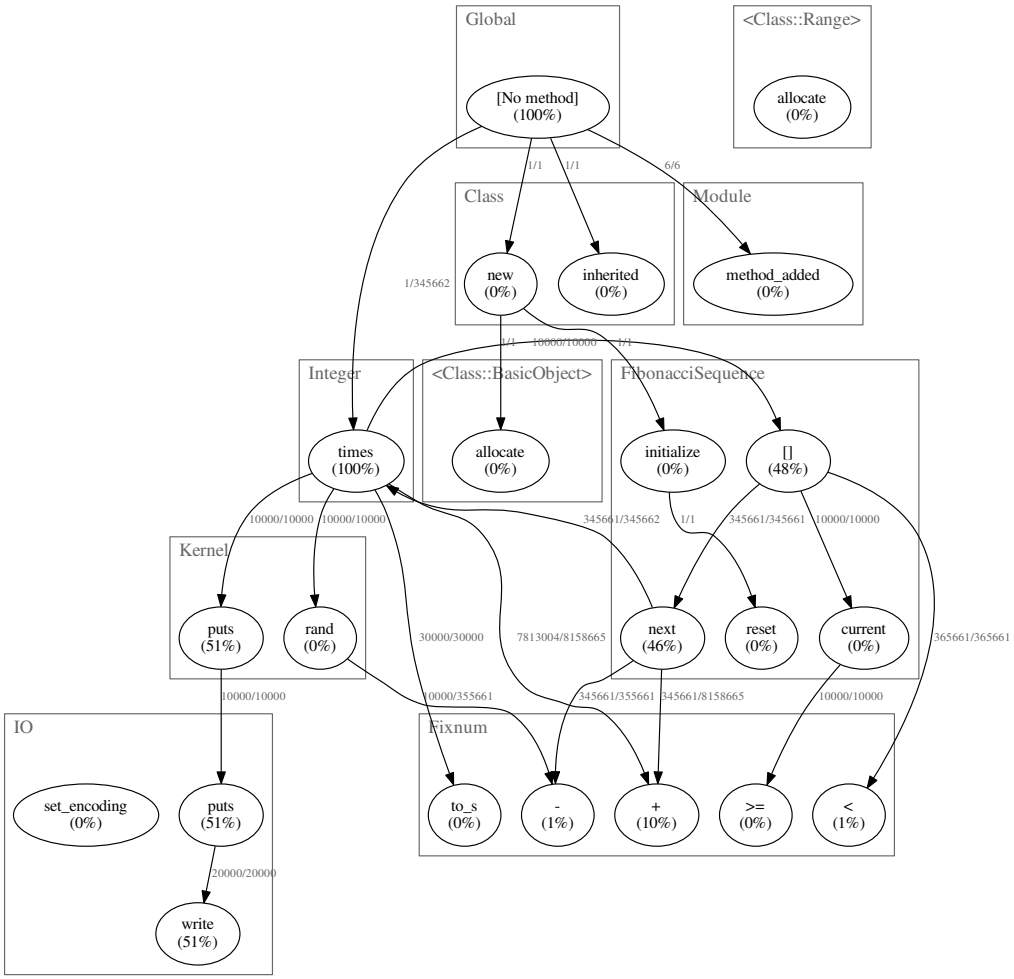
PROFILING FIBONACCIHO POSLOUPNOSTI

Petr Jůda – xjudap00

K profilování Fib. posloupnosti jsem použil nástroj ruby-prof. Programy byly spuštěny 10 000x s úkolem vypočítat různé členy Fibonacciho posloupnosti a vytisknout je na stdout. Při neefektivní implementaci nejsou využívány předchozí členy posloupnosti a každý člen se počítá v cyklu od prvních dvou členů. Z obrázků níže je vidět, že dochází k neefektivnímu volání metody .next, která zabírá téměř polovinu výpočtového času. Při použití optimalizace, tzn. využití již vypočítaných členů uložených v paměti lze běh programu zrychlit až 3x. Dále lze při zrychlení kódu použít např. zkrácené ternární vyhodnocování podmínek typu: return podmínka ? a1 : a2, které Ruby nabízí a mohou urychlovat běh programu.

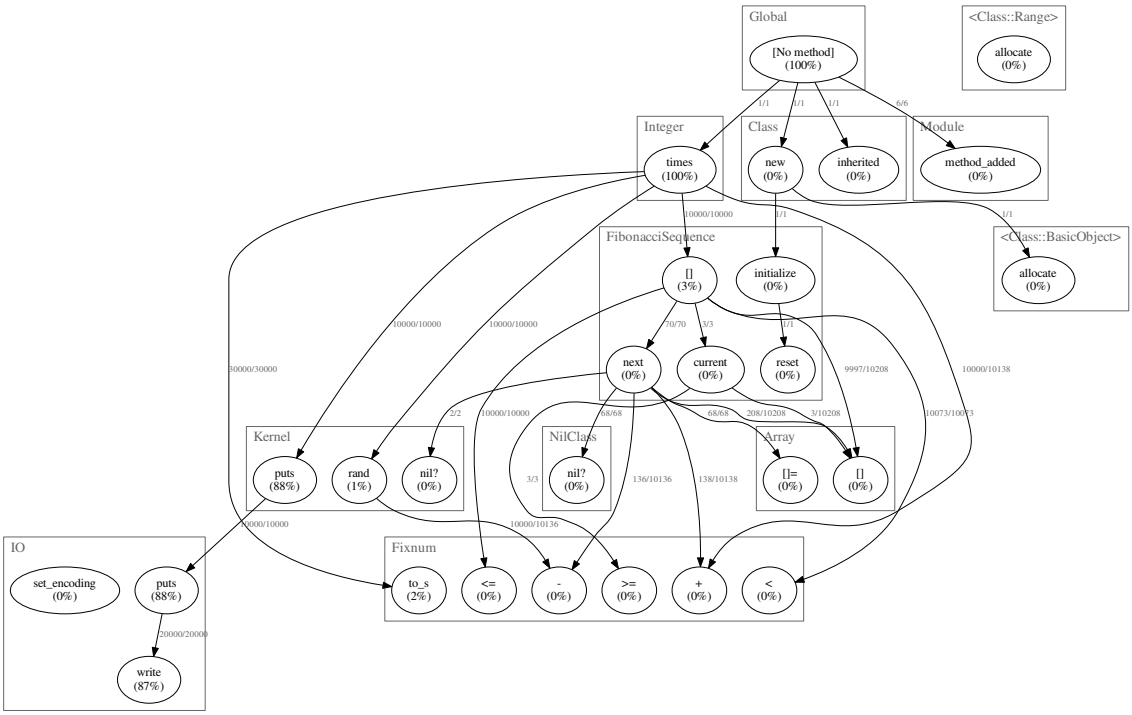
Neefektivní implementace:




```
real    0m2.213s
user    0m0.836s
sys     0m0.064s
petr@petr-VirtualBox:~/Documents/ruby$ time ruby fib-sequence-slow.rb
```

V neefektivní implementaci metoda .[] zavolala 345 661 krát metodu .next, docházelo tak k neefektivnímu vytváření dlouhotrvajících smyček.

Efektivní implementace:



V efektivní implementaci metoda .[] zavolala metodu .next pouze 70 krát, docházelo k využívání již vypočítaných hodnot ukládáním do pole, pokud pro požadovanou hodnotu existoval výsledek v poli byl pouze načten a vypsán.