

Funkcionální jazyky jako Haskell a F# činí kód lépe testovatelným, inherentně vícevláknově bezpečným a snáze analyzovatelným díky omezení vedlejších efektů. Nicméně použití neměnných datových struktur přináší náklady z hlediska efektivity, protože mnoho operací vyžaduje kopírování namísto úprav na místě. Z tohoto důvodu jazyk Koka zavedl nový mechanismus zvaný fully in-place calculus (FIP kalkulus), který umožňuje bezpečné úpravy na konstantní paměti a zároveň se vyhýbá zbytečným alokacím. Správa paměti v Koca a jeho bohatá sada funkcionalit však ztěžují odlišení konkrétních přínosů FIP v oblasti úspory paměti. Cílem této práce je navrhnout minimální funkcionální jazyk jménem StaFip, který podporuje fully in-place aktualizace za pomoci FIP kalkulu a který nevyužívá žádnou správu paměti (garbage collection). Jeho výkon testujeme vůči konvenční implementaci algoritmu quicksort, vkládání do černo-červených stromů a prstových stromů. Výsledky ukazují, že jazyk využívající FIP kalkulus v prostředí bez garbage collection může přinést znatelné zvýšení výkonu a úsporu paměti.