MI-SP-SP-18 (MI-RUN)

Virtuální metody v objektově orientovaných jazycích a jejich implementace, optimalizace vyhledávání metod.

Dynamic dispatch - zavolání správné metody za běhu programu
(nebo rozhodnout v průběhu kompilace)
- mají u polymorfismu (v poli base class může být i child class)

Tabulka virtuálních metod (C++)
- každá třída obsahující virtuální metody má vlastní tabulku
  - kterou vytvořená kompilátorem
  - obsahuje ukazatele na konkrétní implementaci metody
- každý objekt třídy obsahuje ukazatel na takovouto tabulku
  → ví mi se říct, co zavolat

⊖ vyšší spotřeba paměti pro složité hierarchie

Dynamic method lookup (Java, C#)
- ~~každá~~ třída má tabulku pouze svých metod
  - za běhu se v ní hledá daná metoda - pokud se najde, zavolá se
  → v opačném případě hledám v předcích (až do kořene)
- opět se volají jen nad virtuálními metodami (C#: virtual, override)

Single & Multiple Dispatch
- single dispatch = klasické předávání metod v rámci podtřídy
  - známe jednu věc a to třídu objektu
- multiple dispatch - více metod se stejným názvem lišících se v typech argumentů
  - nejedná se o přetěžování vyřešené při kompilaci

Optimalizace vyhledávání metod

globální cache - hash tabulka s klíčem (třída, metoda) a hodnotou address metody
- pevné dané velikosti

Inline cache, Polymorfic inline cache

- většina metod na určitém místě kódu (call sites) je pořád stejná
  → místo vyhledávání se dá skočit kam určitě metoda
- inline přímo v kódu
- v případě, že se neshodují, dělá se lookup

- PIC na jednom místě obsahuje více volání (malý počet)
  - relativně malý počet volání (jednotky)
- call sites: monomorfické (IC), polymorfické (PIC), megamorfické

Devirtualizací

- vyřešení virtualizace při kompilaci
  → nevznikají virtuální tabulky
- velmi náročné algoritmicky