11 Warp

Warp je vysoce výkonná knihovna pro HTTP server napsaná v jazyce Haskell, který je čistě funkcionálním programovacím jazykem. Jak Yesod, webový aplikační framework, tak mighty, HTTP server, jsou implementovány nad Warpem. Podle našich srovnávacích testů propustnosti, mighty poskytuje výkon na stejné úrovni jako nginx. Tento článek vám vysvětlí architekturu Warpu a to, jak jsme dosáhli jeho výkonnosti. Warp může běžet na mnoha platformách, včetně Linuxu, variant BSD, Mac OS a Windows. Pro zbývající část tohoto článku budeme kvůli zjednodušení našeho vysvětlování mluvit jen o Linuxu.

11.1 Síťové programování v Haskellu

Někteří lidé se domnívají, že funkční programovací jazyky jsou pomalé nebo nepraktické. Nicméně, dle našeho nejlepšího vědomí, Haskell poskytuje téměř ideální přístup k síťovému programování.

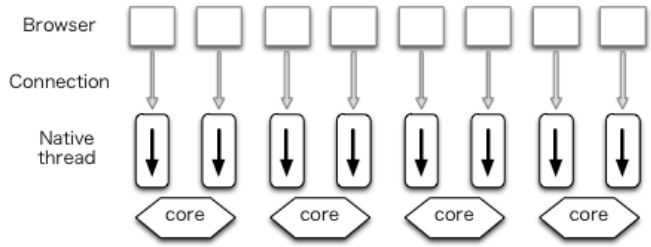
Důvodem je, že Glasgow Haskell Compiler (GHC), vlajková loď mezi Haskell kompilátory, poskytuje lehké a robustní uživatelská vlákna (někdy nazývané zelená vlákna). V této části krátce přezkoumáme některé dobře známé přístupy k síťovému programování na straně serveru a porovnáme je se síťovým programováním v Haskellu. Ukážeme, že Haskell nabízí kombinaci programovatelnosti a výkonnosti, která není k dispozici v jiných přístupech: Vyhovující abstrakce Haskellu umožňují programátorům psát jasný, jednoduchý kód, zatímco sofistikovaný kompilátor Haskell a vícejádrový běhový systém produkují vícejádrové programy, které se vykonávají způsobem velmi podobným nejpokročilejším ručně zhotoveným síťovým programům.

Nativní vlákna

Tradiční servery používají techniku zvanou vláknové programování. V této architektuře je každé spojení zpracováno jedním procesem nebo nativním vláknem (někdy nazývané OS vlákno).

Tato architektura může být dále členěna na základě mechanismu použitého pro vytvoření procesu nebo nativního vlákna. Při použití zásobníku vláken je předem vytvořeno více procesů nebo nativních vláken. Příkladem toho je režim prefork (vytvoření zásobníku vláken/procesů předem) v Apache. Jinak je proces nebo nativní vlákno vytvořeno pokaždé, když se naváže spojení. Toto ukazuje Obrázek 11.1.

Výhodou této architektury je, že umožňuje psát vývojářům jasný kód. Zejména použití vláken umožňuje, aby kód sledoval jednoduchý a známý řídící tok a pouze jednoduché volání procedur pro načtení vstupu nebo odeslání výstupu. Také proto, že jádro operačního systému přiřadí procesům nebo nativním vláknům dostupná jádra, můžeme vyvážit zátížení všech procesorových jader.

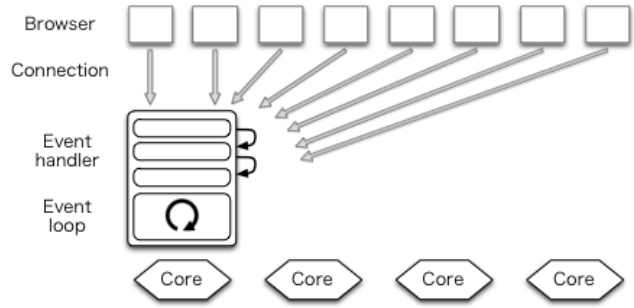


Obrázek 11.1: Nativní vlákna

Jeho nevýhodou je, že nastává velký počet přepínání kontextu mezi jádrem operačního systému a procesy nebo nativními vlákny, což vede k degradaci výkonu.

Událostmi řízená architektura

Ve světě vysoce výkonných serverů bylo programování řízené událostmi nedávným trendem. V této architektuře je více připojení řízeno jediným procesem (Obrázek 11.2). Lighttpd je příkladem webového serveru používajícího tuto architekturu.



Obrázek 11.2: Událostmi řízená architektura

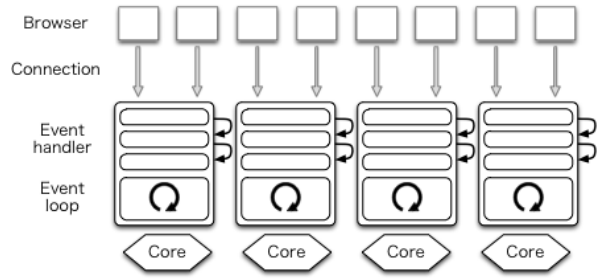
Vzhledem k tomu, že není třeba přepínání procesů, dochází k méně přepínáním kontextu, a výkon je tím vylepšen. To je jeho hlavní výhodou.

Na druhou stranu, tato architektura podstatně komplikuje síťový program. Zejména obrací tok řízení tak, aby událostní smyčka řídila celkové provádění programu. Programátoři proto musí restrukturalizovat svůj program do událostních rutin, z nichž každá provádí pouze neblokující kód. Toto omezení brání programátorům vykonávat I/O operace pomocí volání procedur; místo toho musí být použity složitější asynchronní metody. Podle stejných pravidel, konvenční metody pro zpracování výjimek již nejsou použitelná.

Jeden proces na jedno jádro

Mnozí přišli s nápadem vytvořit n událostmi řízených procesů, které využívají n jader (Obrázek 11.3). Každý proces se nazývá dělník. Mezi dělníky musí být sdílen servisní port. Sdílení portu může být dosaženo pomocí prefork techniky.

V tradičním procesu programování je proces pro nové spojení spuštěn po navázání tohoto spojení. Naproti tomu prefork technika spouští procesy ještě před navázáním nových spojení. I když mají stejný název, tato technika by neměla být zaměňována s režimem prefork serveru Apache.



Obrázek 11.3: Jeden proces na jedno jádro

Webový server, který používá tuto architekturu, je nginx. Node.js používal událostmi řízenou architekturu v minulosti, ale v poslední době také implementoval prefork techniku. Výhodou této architektury je to, že využívá všech jader a zvyšuje výkon. Nicméně, nevyřeší problém programů, které mají špatnou srozumitelnost v důsledku spoléhání se na funkce obsluh a zpětného volání.

Uživatelské vlákna

Problém srozumitelnosti kódu může být vyřešen pomocí uživatelských vláken GHC. Zejména můžeme zpracovat každé HTTP připojení v novém uživatelském vláknu. Toto vlákno je naprogramováno v tradičním stylu použitím logicky blokujících I/O volání. To udržuje program jasný a jednoduchý, zatímco GHC zpracovává složitosti neblokujících I/O a vícejádrovou práci řízení.

Uvnitř GHC multiplexuje uživatelská vlákna přes malý počet nativních vláken. Běhový systém GHC obsahuje vícejádrový plánovač vláken, který může levně přepínat mezi uživatelskými vlákny, protože to dělá bez účasti přepínání kontextu operačním systémem.

Uživatelská vlákna GHC jsou lehké; na moderních počítačích může běžet 100 000 uživatelských vláken. Jsou robustní; dokonce jsou zachyceny i asynchronní výjimky (tato funkce se používá obsluhou časového limitu a je popsána v části 11.2 a v části 11.7). Kromě toho plánovač obsahuje algoritmus vícejádrového vyrovnávání zátěže pro lepší využití kapacity všech dostupných jader.