Úvod

Explicitní model-checking

DIVINE

Komunikační rozhraní

Program DIVINE dokáže zpracovávat stavový prostor ve dvou režimech paralelizace. První z nich je paralelizace ve sdílené paměti, která je v aktuální verzi programu (DIVINE 3.x) upřednostňována. Některé důvody jako například možnost komprese stavového prostoru – a tudíž efektivnější využívání paměti – nebo rovnoměrnější rozvržení pracovní zátěže jednotlivých vláken – což vede k rychlejšímu prohledávání stavového prostoru – jsou popsány v [Vláďova bakalářka] a v [moje bakalářka].

Druhý režim paralelizace je hybridní a zahrnuje práci v distribuované i ve sdílené paměti. Tento režim pochází ze starší verze programu (DIVINE 2.x) a oproti původní verzi nebyl nikterak vylepšován (až na malou optimalizační změnu). Hybridní paralelizmus je realizován tak, že každý stav ze zpracovávaného stavového prostoru je staticky přiřazen některému vláknu na některé samostatné výpočetní jednotce pomocí hašování [odkaz na hash]; v DIVINE se používá konkrétně Spooky Hash [odkaz na Spookyhash]. Jako komunikační vrstva je použit standard MPI[odkaz na MPI], konkrétně implementace OpenMP[odkaz na OpenMPI]

Hlavní nevýhodou původní implementace hybridního paralelizmu bylo statické rozdělení stavů nejen mezi jednotlivé výpočtní stroje, ale i mezi jednotlivá vlákna. Toto rozdělení má kromě nevýhody v potenciálně nerovnoměrném rozložení práce mezi jednotlivá vlákna i nevýhodu v nemožnosti použít aktuální implementaci komprese paměti[Vláďova bakalářka].

Již v průběhu vytváření režimu paralelizace ve sdílené paměti bylo zřejmé, že by bylo možné upravit stávající hybridní režim tak, aby v rámci jednotlivých výpočetních jednotek byl použit režim paralelizace ve sdílené paměti, kdežto pro rozdělení práce mezi výpočetní jednotky by nadále používalo statické rozdělování stavů na základě haše. Tento režim, pracovně nazvaný dvouvrstvá architektura, ovšem z důvodu upřednostnění jiných úkolů nebyl nikdy realizována.

V současné době se pracuje na nové verzi programu DIVINE, přičemž součástí změn je i úprava modelu paralelního zpracování stavového prostoru a zavedení jednotného režimu paralelizace pomocí dvouvrstvé architektury. Z tohoto důvodu bylo zvažováno, jestli by jiná komunikační vrstva nebyla jednodušší na použití a jestli by nebyla efektivnější při práci s pamětí. Další věc, kterou jsme zvažovali, byla co nejmenší závislost na externích knihovnách.

Před volbou vhodného komunikačního rozhraní bylo potřeba definovat, v jakém prostředí bude program DIVINE spouštěn, a tedy jaká jsou hlavní krutéria výběru. Očekáváme, že

MPI

Vlastnosti

Rozhraní v DIVINE

Boost - asio

Vlastnosti

Vlastní implementace

Další možností je vlastní implementace komunikačního rozhraní pro DIVINE, které by používalo BSD sockety[odkaz na BSD sockety], které jsou v zahrnuty v POSIX standardu[odkaz na POSIX]. Vlastní

implementace nepřidává žádnou závislost na externí knihovně a protože jsou BSD sockety v podstatě standardem pro síťovou komunikaci[dodat odkazy na pojednávající články], lze předpokládat, že výsledný kód bude možné bez větších změn použít i na operačních systémech, které nevycházejí z filozofie systému UNIX.

BSD sockety

BSD sockety nabízí tři druhy možných spojení – spojité, nespojité a ???. Ke každému druhu spojení je třeba přistupovat jinak a bylo potřeba na základě jejich vlastností vybrat takové spojení, které by nejlépe vyhovovalo požadavkům programu DIVINE.

Všechny tři způsoby mají stejný způsob adresace. Je potřeba uvést IP adresu[IP] stroje v síti, se kterým bude vedena komunikace, a dále je potřeba uvést port[port], kterým se specifikuje, který program, případně službu, má operační systém spojit s danou žádostí o komunikaci. Další pokračování komunikace se liší právě na základě druhu spojení, proto je proberu každý zvlášť.

IP adresy (bez rozlišení mezi IPv4 a IPv6 [odkaz]) mají svůj přesně definovaný formát a stoje dostávají adresu při přihlášení do sítě dle nastavení sítě. Protože IP adresy jsou pro člověka obtížně zapamatovatelné, je možné přidělit v rámci sítě strojům jednoznačná jména, která se pomocí služby DNS[DNS] v síti přeloží právě na IP adresu.

Port je číslo v rozsahu 1 – 65 536, pomocí kterého se v operačním systému adresují služby dostupné přes síťové rozhraní. Čísla v rozsahu 1 – 1024 jsou rezervovány pro známé síťové služby, jako například služba pro posílání e-mailů (SMTP protokol [SMTP] – port 25), služba webových stránek (HTTP protokol[HTTP] – port 80) a mnohé další. Proto moderní systémy obvykle neumožňují používat tyto porty pro příchozí komunikaci bez administrátorského oprávnění[linux][windows].

Spojité

Spojité sockety jsou známé svým použitím v implementaci protokolu TCP[odkaz]. Komunikace tímto způsobem probíhá tak, že výpočetní stroj, který očekává příchozí spojení (dále jako server), otevře na své straně na určeném portu socket[bind,listen], skrze něhož hodlá zpracovávat příchozí spojení. Výpočetní stroj, který chce se serverem komunikovat (dále jako klient), požádá[connect] o spojení na server na předem definovaný port, následkem čehož je server notifikován a může přijmout[accept] příchozí spojení. Přijetím spojení se vytvoří na straně serveru další socket, který pak slouží jako jeden konec obousměrné komunikace, a obdobně získá klient na své straně socket pro komunikaci se serverem. Pomocí získaného socketu lze obousměrně komunikovat jak standardními POSIXovými funkcemi[read,write] tak funkcemi specifickými pro práci se sockety[send,recv], případně měnit vlastnosti socketu[setsockopt].

Pozorného čtenáře by mohlo napadnout, jak je v operačním systému adresováno právě získané spojení. Adresace je vyřešena tak, že jak na straně klienta tak na straně serveru se vytvoří nové sockety, které jsou navázány na nějaký port, který vybírá operační systém z volných portů. Není proto na žádné straně nutné (a ani možné) specifikovat konkrétní port pro dané spojení.

Data se mezi dvěmi konci spojení posílají pomocí paketů[packet]. Spojité sockety garantují{footnote: Je garance opravdu možná?}, že spojení bude udržováno, dokud ho jedna z komunikujících stran neuzavře. Dále garantují, že odeslaná data dojdou a že dojdou ve správném pořadí, v jakém byly odeslány. Pro implementace všech těchto garancí se využívá hlavička paketů[hlavička paketu], ve které se definují věci jako pořadí zprávy, kontrolní součet dat posílaných ve zprávě a mnohé další. Pro udržování spojení se pak používá speciální paket, který si musí komunikující strany v pravidlených intervalech posílat, ovšem je možné tento speciální paket vložit do běžné datové zprávy.

[obrázek hlavičky paketu]

Níže je sepsán seznam vlastností spojitého komunikačního kanálu, které jsou z hlediska použití programem DIVINE klíčové:

- udržování spojení

- možnost mít více kanálů mezi výpočetními stroji
- garance doručení nepoškozených dat
- možnost detekce "živosti" komunikačního partnera

Udržování spojení

Více kanálů

Garance doručení nepoškozených dat

Detekce "živosti" komunikačního partnera

Nespojité

???(třetí způsob)

Vlastní definice workflow

XXX

Popis komunikačního protokolu

Bezpečnost

Rozhraní pro distribuované procházení grafu

Experimentární porovnání

Závěr