IN-MEMORY DATABASE MANAGEMENT SYSTEM

Petr Pehal
Master Degree Programme (3), FIT BUT
E-mail: xpehal00@stud.fit.vutbr.cz

Supervised by: Marek Rychlý
E-mail: rychly@fit.vutbr.cz

Abstract: The focus of this paper is a proprietary database interface for management tables in main memory. At the beginning, there is given a short introduction to the information and control system for which the database was created. Then the main aspects of this database interface are discussed. Further the work aims at the specification and design of modifications, extensions, and optimalizations that the in-memory database management system requires.

Keywords: In-memory databases, power grid management, proprietary in-memory database interface, embedded database, in-memory index structures, sharing main memory database tables

1 ÚVOD

Systémy pro řízení báze dat představují v dnešní době klíčovou součást téměř všech řídících a informačních systémů. Nalezme však také případy, kdy rychlost odezvy systému řízení báze dat nemusí být pro řídící procesy dostatečná. Typicky se jedná o procesy podílející se na zpracování velkého množství dat v reálném čase. Právě tomuto problému čelí řídící a informační systém RIS od společnosti ELEKTROSYSTEM, a.s [1], který je již přes 20 let využíván k řízení energetické sítě v České i Slovenské republice. Pro splnění vysokých výkonnostních nároků využívá společnost ELEKTROSYSTEM vlastní proprietární databázi uloženou a spravovanou v operační paměti.

Tato práce je zaměřena na rekonstrukci a modernizaci tohoto specializovaného systému řízení báze dat v operační paměti. Nejdříve je představen současný stav a následně je popsán návrh významnějších změn.

2 PAMĚŤOVÁ DATABÁZE V SYSTÉMU RIS

Systém RIS se svojí funkčností řadí mezi *SCADA* systémy [2]. Primárním úkolem je tedy sběr a zpracování údajů z telemetrických zařízení, která zprostředkovávají informace o stavu energetické sítě. Na straně systému jsou tyto informace rozebírány, upravovány, archivovány a prezentovány uživateli. Úlohu zpracování a úpravy dat v systému obstarává řídící proces zvaný *Rdb*. Na tento proces jsou kladeny značné výpočetní nároky, jelikož se v běžném provozu energetických sítí musí vypořádat až s několika tisíci změnami měřených veličin během jedné sekundy. Rdb pak pro každé přijaté měření provádí příslušné akce a odesílá změny dalším procesům v systému.

Datová základna Rdb je vystavěna na proprietárním paměťovém databázovém rozhraní označovaném jako *dbi*. Dbi bylo původně vytvořeno speciálně pro tento řídící proces. Následně se ale značně rozšířilo a dnes představuje jeden ze základních stavebních kamenů celého systému.

Dbi je knihovna jazyka C, která umožňuje definovat, vytvářet a spravovat relační tabulky v operační paměti. Tabulky vznikají přímo v paměti daného procesu, který zažádal o jejich vytvoření. Kromě toho je možné k tabulkám přistupovat také vzdáleně z jiných procesů s využitím proprietární meziprocesové komunikace. Oba typy přístupů jsou dostupné prostřednictvím uniformního API. N-tice

jsou reprezentovány strukturami jazyka C. V případě lokálních tabulek je pak možná práce přímo s těmito strukturami, což přináší velmi vysokou efektivitu. Perzistence tabulek je zajištěna pomocí techniky ukládání snapshotů.

Využití vlastní paměťové databáze má za důsledek vysokou rychlost zpracování stavu energetické sítě, což představuje jeden z klíčových prvků úspěšnosti celého systému. Kapacita zpracovávaných dat během let provozu systému však značně vzrostla a pomalu se začíná blížit limitům současného řešení. Primárním cílem úprav dbi je tedy práce s většími datovými objemy. Pod pojmem „větší datové objemy“ chápeme v souvislosti s energetickou sítí desítky až stovky miliónu záznamů.

3 NOVÉ DATABÁZOVÉ ROZHRANÍ

Rekonstrukce databázového rozhraní pokrývá hned několik různorodých oblastní. Zde budou stručně představeny některé z nejvýznamnějších úprav.

3.1 ZÁKLADNÍ DATOVÉ STRUKTURY

Degradující efektivita spojená s rostoucím množstvím objektů je zapříčiněna návrhem datových struktur. Jako primární datovou i indexovou strukturu využívá současné dbi dvouúrovňově segmentované pole. Dbi bylo původně navrhováno pro počty prvků v řádech desítek tisíc. Zde bylo tedy pole vhodným řešením, přinášejícím vysokou rychlost přístupu. Pro objemnější data však režie spojená s přesouváním prvků v indexových strukturách výrazně roste. Řešením tohoto problému je přiblížení implementace indexů stromovým strukturám. Nová indexová struktura je reprezentována tříúrovňovým polem, kde segmenty na jednotlivých úrovních mají vždy velikost mocniny dvou. Díky tomuto faktu je možné provádět výpočet pozice v každé úrovni pomocí rychlých bitových operací. Prvky v jednotlivých segmentech musí být uspořádány, avšak ne všechny pozice v segmentu musí být obsazeny. Tím je minimalizována režie spojená s přesunem prvků při modifikačních akcích. Neobsazení všech pozic vyžaduje doplnění implementace slučování a vkládání segmentů. Jedná se tedy o setříděný index, ve kterém vyhledávání probíhá půlením intervalů. Indexová struktura může být také volitelně doplněna o hashovací tabulku, kde jsou uloženy odkazy na jednotlivé objekty. Tento doplňkový mechanizmus je výhodný u indexů vybudovaných nad textovými řetězci, kde vyhledávací akce výrazným způsobem převažují nad akcemi modifikačními. Přesně tomuto popisu odpovídá databáze objektů Rdb, kde vyhledávání dle jména probíhá při každé přijaté změně z telemetrií a naopak vkládání, mazání či přejmenovávání veličin je prováděno pouze několikrát denně správci systému.

3.2 UKLÁDÁNÍ SNAPSHOTŮ

Další oblastí, kde výkonnost dbi při zpracování většího počtu prvků zaostává, je ukládání snapshotů. Důvodem je implementace snapshotu formou celistvého souboru s jednorázovým zápisem. Formát snapshotu byl proto rozdělen do více nezávislých souborů, což umožňuje částečné paralelní zpracování. Především ale rozdělení otevřelo cestu nespojitému zápisu jednotlivých struktur. Při uložení snapshotu jsou na disk zapsány vždy pouze modifikované segmenty. V případě indexových struktur je však nutné řešit problémy spojené se správným určením diskové pozice daného segmentu. Musíme přitom dbát na zachování efektivity při načítání i ukládání a nesmíme zapomenout ani na minimalizaci případné fragmentace. Aby bylo možné splnit oba relativně protichůdné požadavky, musíme připustit situaci, že pořadí ukládání segmentů na disk může být rozdílné od skutečného pořadí segmentů ve struktuře. Každý segment si tak uchovává svoji diskovou pozici. Nově vloženému segmentu je přidělena pozice vždy na konci souboru. K přesunu na disku dochází pouze pokud je některý segment odstraněn. V tomto případě převezme pozici segmentu odstraněného poslední segment uložený na disku. Aby bylo možné provést rekonstrukci struktury z diskového souboru, uchovává si každý na disku uložený segment skutečnou pozici svého následníka v indexové struktuře.

3.3 VLASTNÍ SPRÁVA PAMĚTI DATOVÝCH TYPŮ PROMĚNNÉ VELIKOSTI

Nový mechanizmus ukládání snapshotů přináší i další komplikaci. Tu představuje uchovávání hodnot atributů proměnné velikosti. V původním snapshotu byly tyto hodnoty zapsány za obsahem datových struktur sekvenčně v pořadí, v jakém příslušely k daným databázovým objektům. Zápis i rekonstrukce těchto hodnot ze snapshotu pracovaly správně zásluhou jednorázového provedení těchto činností. Při použití ukládání po segmentech je proto nutné zavést vlastní správu paměti pro hodnoty datových typů proměnné velikosti (pro lepší srozumitelnost budeme tyto hodnoty dále označovat jako *řetězce*).

S diskem opět pracujeme po segmentech. Vyžadujeme tedy, aby řetězce ležely v rámci segmentu blízko u sebe a předcházeli jsme tak fragmentaci. Od operačního systému požadujeme spojité paměťové bloky o velikosti segmentu. Nad těmito segmenty pak probíhá dynamické přidělování paměti pro řetězce. Paměť je vždy přidělována i uvolňována se stanoveným minimálním krokem. Samotná správa je řízena pomocí front volných paměťových bloků. Pro každý násobek minimálního kroku až po hranici určenou velikostí segmentu vyhradíme jednu frontu. Při požadavku na alokaci nového bloku paměti jsou nejdříve vzestupně zkontrolovány všechny příslušné fronty. Pokud je nalezen volný blok, je jeho velikost upravena a přebývající část bloku je opět vložena do patřičné fronty. Jestliže již neexistuje žádný volný blok, proběhne žádost o nový segment od operačního systému. V případě uvolňování paměťového bloku je před uložením do fronty zkontrolováno, zda není možné sloučit blok s volným následníkem či předchůdcem.

3.4 SNÍŽENÍ ZÁTĚŽE S VYUŽITÍM REPLIKACE

Na závěr ještě nastíníme optimalizační techniku, která umožní výrazným způsobem odlehčit zatížení řídícího procesu Rdb. Využijeme zde principu replikace. Kromě neustálého zpracovávání aktuálních informací od telemetrií je Proces Rdb zaměstnáván také rozesíláním změn jiným procesům a dotazy směřujícími od dalších procesů na obsah databáze. Časově kritické je přitom pouze zpracování na straně Rdb. To, zda ostatní procesy obdrží změny například o sekundu později, je z pohledu řízení energetické sítě irelevantní. Využije se tedy zmiňované replikace a vytvoří se pomocný proces, který bude vlastnit kopii databáze energetických veličin. Rdb pak bude odesílat změny pouze tomuto pomocnému procesu. Stejně tak veškeré dotazy budou směrovány na tento pomocný proces. V případě potřeby nám nic nebrání jít s touto myšlenkou ještě dále a využít mechanizmu replikace databáze do pomocného procesu opakovaně. Tentokrát můžeme například replikovat databázi Rdb na některou z klientských stanic, kde běží procesy vyžadující vysokou rychlost odezvy na dotazy.

4 ZÁVĚR

V tomto příspěvku bylo představeno praktické využití databáze v operační paměti pro řešení zpracování dat v reálném čase v oblasti energetického průmyslu. Na začátek byl nastíněn řídící systém využívající tuto databázi společně s požadavky, které jsou na něj kladeny. Následně byly charakterizovány některé z významnějších změn prováděných v rámci modernizace tohoto specifického systému řízení báze dat. Nakonec byla stručně popsána jedna z nově zaváděných optimalizačních metod pro urychlení běhu celého řídícího systému.

REFERENCE

[1] Elektrosystem: *Řídící a informační systém RIS* [online]. [cit. 2013-02-23]. Dostupné z WWW: http://www.esys.cz/ris.php

[2] NATIONAL COMMUNICATIONS SYSTEM: *Supervisory Control and Data Acquisition (SCADA) Systems* [online]. Arlington, 2004. [cit. 2013-02-24]. Dostupné z WWW: http://www.ncs.gov/library/tech_bulletins/2004/tib_04-1.pdf