Návrh consistency control algoritmů

Near strong consistency control, MMWP a MMWP batch

Autor:
Pavel Bžoch
pbzoch@kiv.zcu.cu

Návrh near strong consistency control algoritmu

Near strong consistency je algoritmus pro udržení konzistentní cache, který má/vyžaduje následující vlastnosti:

- Server je bezstavový (stateless). V našem případě to znamená, že server neposílá klientům žádné informace o změněných souborech. V KIV-DFS tento přístup šetří čas a prostředky, protože by bylo obtížné udržovat seznam všech klientů majících daný soubor ve své cache na všech serverech, který tento soubor poskytují.
- Pro udržení konzistentní cache se použije tzv. pull přístup. Klient si v tomto přístupu sám stahuje změněná data do své cache.
- Nová verze souboru se kontroluje vždy při požadavku na soubor.
- Tento přístup splňuje požadavky na near strong consistency, takže s touto úrovní konzistentnosti budeme pracovat.

Jak už výčet vlastností napovídá, kontrola konzistentnosti souboru se provádí vždy, když je soubor požadován. Tento přístup není nový, používá se např. v NFS. Nevýhodou tohoto přístupu je fakt, že klientská aplikace musí mít stálou konektivitu k serveru, aby bylo možné verifikaci použít. Pokud není možné zajistit stálou konektivitu, ani požadovaná úroveň konzistentnosti není garantována. Tento přístup navíc není vhodný pro mobilní klienty, protože je neustálé zjišťování a verifikace souborů je energeticky náročná (nutnost vysílat a přijímat data). Pro potřeby mobilních klientů byly tedy navrženy další algoritmy, které jsou popsány dále. Při návrhu těchto algoritmů bylo již počítáno s tím, že budou použity v mobilních aplikacích.

Návrh MMWP consistency control algoritmu

MMWP (Median of Minimal Periods) je algoritmus pro udržení konzistentní cache, který má/vyžaduje následující vlastnosti:

- Server je bezstavový (stateless). V našem případě to znamená, že server neposílá klientům žádné informace o změněných souborech. V KIV-DFS tento přístup šetří čas a prostředky, protože by bylo obtížné udržovat seznam všech klientů majících daný soubor ve své cache na všech serverech, který tento soubor poskytují.
- Pro udržení konzistentní cache se použije tzv. pull přístup. Klient si v tomto přístupu sám stahuje změněná data do své cache.
- Pro kontrolu nových verzí souborů bude použit tzv. TTL (Time-To-Live) pro každý cachovaný soubor. TTL je doba, po kterou je soubor v cache považován za konsistentní. Jakmile tato doba vyprší, je nutné obnovit (znovu se zeptat) na stav (verzi) souboru.
- TTL přístup splňuje požadavky na weak consistency, takže s touto úrovní konzistentnosti budeme pracovat.

Problém, který můžeme vypozorovat, je správné nastavení doby TTL. Pokud bude tato doba příliš krátká, bude docházet k vyššímu zatěžování komunikačního média (počítačové sítě) a metadata serveru. Pokud bude doba příliš dlouhá, může dojít k tomu, že uživatel bude pracovat s cachovanými daty, které nejsou konzistentní, protože došlo k jejich změně na straně serveru. Nastavení správné hodnoty TTL je tedy zásadní pro správnou funkci MMWP.

Při pozorování přístupu uživatelů k datům (souborům) jsme zjistili, že některé soubory jsou měněny často a některé nejsou měněny vůbec. U měnících se souborů lze vypozorovat závislost periody zápisu na počtu zápisů (konkrétněji závislost mediánu z minimálních dob zápisu na počtu zápisů). Tato závislost je zobrazena v tabulce 1. Veškerá naměřená data pochází z logu lokálního AFS, který byl pořízen v týdnu od 27.1 do 3.2 2013.

TABULKA 1. ČASOVÉ ÚSEKY ZÁPISŮ NOVÉHO OBSAHU SOUBORŮ

Počet zápisů	Počet souborů	Průměrná minimální doba zápisu [ms]	Medián z minimálních dob zápisu [ms]
2	1691	1 031 082	128 000
3	30	525 226	62 500
4	21	2 464 190	51 000
5	5	2 882 500	43 000
6	13	1 769 429	49 500
7	7	400 125	33 400
8	19	30 200	31 000
9	9	56 222	22 000
10 to 15	72	80 958	30 000
16 to 20	22	78 217	30 000
20 to 30	44	31 089	17 000
30 to 50	16	9 706	1 000
50 to 100	48	11 483	1 000
100 to 500	47	2 041	1 000
500 <	23	1 000	1 000

Na základě tohoto pozorování jsme vybudovali algoritmus, který každému souboru přiřadí hodnotu TTL podle počtu zápisů do tohoto souboru. Počet zápisů (WRITE_HITS) je poskytován ke každému

souboru jako součást metadat daného souboru. Na straně serveru je tento čítač periodicky dělen dvěma, aby se předešlo tzv. ageingu. Při přiřazování hodnoty TTL je soubor přidělen do jedné z pěti skupin podle počtu zápisů (viz tabulka 2). Tyto skupiny respektují pozorování z tabulky 1.

TABULKA 2. ROZDĚLENÍ SOUBORŮ DO SKUPIN PODLE POČTU ZÁPISŮ

Počet zápisů	Skupina
1	G_1
2-3	G_2
4-6	G_3
7-30	G_4
30+	G_5

Přiřazení časů TTL je uskutečněno na základě měření. Cílem provedených měření bylo nastavit hodnoty TTL tak, aby byly co nejvyšší a zároveň, aby všechny přistupované soubory byly v konzistentním stavu. Přidělení časů TTL k jednotlivým skupinám je vidět v tabulce 3.

TABULKA 3. PŘIŽAZENÍ TTL K JEDNOTLIVÝM SKUPINÁM

Skupina	TTL [s]
G_1	300
G_2	90
G_3	20
G_4	20
G_5	10

Jakmile víme čas TTL pro každý soubor, můžeme spustit MMWP consistency control.

Návrh MMWP batch consistency control algoritmu

Tento algoritmus je postaven na stejném principu jako předchozí algoritmus. Snaží se ale eliminovat jednu nevýhodu předchozího algoritmu. Tato nevýhoda spočívá v tom, že se na každý soubor v cache ptáme separátně. Tudíž pro verifikaci konzistentnosti musíme vytvořit a vyslat zprávu, která obsahuje jméno souboru a číslo jeho cachované verze. Od metadata serveru se nám vrátí informace, jestli máme konzistentní soubor nebo si musíme stáhnout novou verzi.

Tento přístup vyžaduje pro každé ověření otevření nového spojení a čekání na odpověď. Pro ušetření těchto nákladů lze ověření provádět dávkově. Jelikož jsme souborům přiřadili čas TTL podle příslušnosti do některé ze skupin, můžeme ověřit konzistentnost všech souborů v dané skupině najednou. Při tomto přístupu se tedy vytvoří jedna zpráva, ve které jsou všechny soubory z dané skupiny včetně čísla jejich verze. Jako odpověď od serveru dostaneme jména souborů, které byly změněny a které je potřeba do cache stáhnout.

Tento přístup je také šetrný k bateriím v mobilních zařízeních, protože pro ověření celé řady souborů stačí odeslat a přijmou jedinou zprávu.