Optymalizacja programów Open-Source

Profilery wysokiego poziomu
część 2

Krzysztof Lichota
lichota@mimuw.edu.pl

gprof

gprof

- Pomiar działa na zasadzie instrumentacji kompilowanego kodu (wejścia i wyjścia z funkcji) oraz okresowego sprawdzania stosu (co określony czas działania programu, żeby przypisać czas)
- Program musi być zrekompilowany z odpowiednią opcją gcc
- Uruchomiony program generuje plik ze statystykami (gmon.out)
- Program gprof analizuje plik ze statystykami i przedstawia wyniki.
- Narzut – niewielki (ale większy niż np. Google profiler)

Zalety gprof

- Dokładnie zlicza wejścia i wyjścia z funkcji
- Dostępny na większości instalacji Linuksa
- Można włączyć profiling tylko dla niektórych modułów programu (przy kompilacji)
- Stanowi standard, niektóre inne narzędzia generują wyniki w jego formacie, więc narzędzia obsługujące gprof działają także tam.
- Informacje z profilowania mogą być użyte do sterowania rekompilacją

Wady gprof

- Wymaga rekompilacji programu i bibliotek używanych przez program
- Nie pokazuje, kiedy program śpi (np. z powodu wejścia/wyjścia albo swapowania)
- Czas jest zliczany przez próbkowanie i przypisywany do funkcji na podstawie liczby wystąpień (z założeniem, że każde wykonanie funkcji trwa tyle samo), więc jest to niedokładne

Użycie gprof - kompilacja

- Trzeba skompilować i zlinkować program z opcją -pg (przydaje się też -g do informacji o liniach w programie i -fprofile-arcs do instrumentacji bloków prostych)
- W przypadku systemu budowania autotools można użyć: CFLAGS="-pg -O0" LDFLAGS=-pg CPPFLAGS="-pg -O0" ./configure
- Lub jeśli jest dostępne to: ./configure –enable-profile
- Żeby uzyskać dobre wyniki należy zrekompilować również biblioteki (niektóre już są, np. libc_p.a).

Użycie gprof - uruchomienie

- Program uruchamia się normalnie, bez dodatkowych opcji
- Przy zakończeniu program zapisuje plik gmon.out w bieżącym katalogu
- Jeśli nie ma się praw zapisu do bieżącego katalogu, plik nie zostanie zapisany
- Jeśli program zostanie zakończony w niespodziewany sposób (nie przez exit()), to plik też nie zostanie zapisany

Użycie gprof – wyświetlanie wyników

- Zwykłe wywołanie: gprof opcje program-wykonywalny gmon.out >plik.txt (generuje tekstowe podsumowanie z czasem wykonywania dla poszczególnych funkcji i drzewem wywołań)
- Opcja –annotated-source wyświetla kod programu z anotacją o czasie przebywania w danym miejscu
- Opcje --exec-counts -l – pokazują ile czasu program spędził w każdym bloku prostym
- Opcja –line pokazuje liczbę wywołań dla każdej linii programu

Użycie gprof – wyświetlanie wyników (2)

- Istnieją narzędzia do graficznego przedstawiania wyników gprof (np. kprof dla KDE)
- Mogą one wyświetlać wyniki również innych profilerów używających formatu gprof

Inne przydatne rzeczy w gprof

- Ponieważ dane są zbierane przez próbkowanie, są niedokładne. Można skumulować dane z kilku uruchomień za pomocą opcji -s (dokładny opis w info dla gprof).

Rekompilacja z profilingiem

- Instrumentacja gcc do profilowania (opcja -fprofile-generate) może służyć również do podpowiedzenia kompilatorowi jak optymalnie kompilować przy powtórnej kompilacji.
- Więcej na zajęciach z profilowania niskiego poziomu.

bootchart

bootchart

- Zbiera dane podczas uruchomienia i generuje wykres obrazujący zachowanie systemu w czasie profilowania
- Zbiera informacje (z /proc) o
  - zużyciu CPU (w tym o czasie zużytym na I/O)
  - stopniu użycia dysków
  - uruchomionych programach i co robią (używają CPU, czekają na I/O).
- Uruchomienie następuje w momencie uruchomienia systemu (przez modyfikację initrd)
- Zatrzymanie poprzez uruchomienie usługi bootchart-stop

bootchart (2)

- W initrd uruchamiany jest proces, który co jakiś czas budzi się i zbiera informacje z /proc
- Narzut niewielki, ale jeśli jest dużo procesów może się zwiększyć (informacje z /proc są zbierane dla wszystkich procesów)

Zalety bootchart

- Profiling od momentu uruchomienia systemu
- Przejrzyste, graficzne przedstawienie wyników
- Zbiorcze przedstawienie tego, co się dzieje w systemie
- Możliwość użycia do profilowania również innych fragmentów działania systemu niż start
- Pokazuje jakie procesy są uruchomione i kiedy oraz kiedy śpią
- Pokazuje środowisko w którym został uruchomiony (wersja jądra, opcje, itp.)
- Stosunkowo łatwo można go modyfikować

Wady bootchart

- Nie pokazuje zużycia pamięci
- Z powodu próbkowania może nie pokazać krótko żyjących procesów

Na co należy zwrócić uwagę

- Procesy, które nie działają nie są pokazywane na wykresie, tak samo krótko żyjące procesy, takie same procesy mogą zostać sklejone
- Generowanie wykresu zajmuje dużo procesora i może zaburzyć dalsze działania
- Czas jest liczony od momentu uruchomienia bootchart, czas na załadowanie jądra i rozpakowanie go (oraz czas uruchomienia BIOS-u) nie jest wliczany
- Domyślnie dane, z których był wygenerowany wykres są wyrzucane

Instalacja bootchart

- Zainstalować paczkę z dystrybucji
- Uaktualnić initrd (np. w Ubuntu „update-initramfs”).

Bibliografia

- http://www-128.ibm.com/developerworks/linux/library/l-gr
- info:gprof
- http://bootchart.org/
- http://docs.freebsd.org/44doc/psd/18.gprof/paper.pdf
- http://sourceware.org/binutils/docs/gprof/index.html