Propozycje projektów Gemius SA w ramach ZPP na MIMUW

1. TEXTSQL

Silnik sql-owy oparty o pliki tekstowe z optymalizacją zapytań. Do programu podajemy zapytanie "sql-owe", pliki tekstowe z danymi oraz opis tych plików.

Opis plików składa się np. z:
- opisu kolumn
- informacji w jaki sposób dane w pliku są posortowane
- ile jest szacunkowo wierszy w plikach

Na podstawie takich danych program powinien zoptymalizować zapytanie i je wykonać.

2. REMOTE SCREENS

Program wpomagający pracę z programami ssh i screen.

Program uruchomiony na jednym komputerze zarządza screenami na wielu innych komputerach poprzez protokół ssh, z wygodnym interfejsem tekstowym/graficznym.

Podstawowe funkcje:
- logowanie do zdefiniowanych komputerów, identyfikacja uruchomionych screen-ów
- uruchomienie nowego screena na wybranym komputerze
- grupowanie uruchomionych screen-ów (np. w formie drzewa)
- podgląd zawartości screen-ów
- wygodny interfejs

3. Dodanie możliwośći zakładania więcej niż 32 tys baz danych w mysql

Każda baza danych w mysql przechowywana jest w osobnym katalogu. Większość systemów plików nie potrafi założyć więcej niż 32 tyś z kawałkiem katalogów w jednym katalogu. Tu mogą być dwa rozwiązania - jedno to przerobić nieco system zarządzania katalogami w mysql'u, drugie to zrobić jakąś nakładkę na filesystem (np. używając fuse), która pozwoli obejść to ograniczenie.

4. Integracja python-apache

Alternatywa dla modułu "mod-python" do Apache'a. Próba zrobienia czegoś bliżej PHP'a. Czyli moduł zajmowałby się sesjami, cookiesami, get'ami, post'ami itp. - tak jak w PHP. Pliki pewnie podobne do php'owych, tylko, że językiem byłby znacznie lepszy (moim zdaniem) Python.

5. Kqueue na Linuxa/Epoll na FreeBSD

Fajnie by było przysłużyć się światu w celu unifikacji rozwiązań problemu "c10k". Tu albo należałoby do Linux'a dodać "kqueue" - rozwiązanie popularne w systemach BSD (włącznie z Mac OS X), albo alternatywnie do systemów BSD (głównie FreeBSD) dodać linux'owego epoll'a.

6. System komunikacji - jeden nadawca / wielu odbiorców (coś jak multicast)

Fajnie by było mieć taki system (biblioteka/moduł kernela), który pozwala rozsyłać informacje od jednego nadawcy do wielu odbiorców(ustalonej przez nadawcę liczby odbiorców). Powinno to być trochę podobne do streaming'u, przy czym znam liczbę odbiorców i wymagam, aby wszyscy otrzymali moje dane. Na niskim poziomie oczywiście w celu uzyskania sensownej wydajności powinna być to komunikacja UDP z użyciem broadcast'ów, lub multicast'ów - w końcu wszystkim wysyłamy to samo, więc nie chcemy tego wysyłac n razy.

7. Automatyczna kategoryzacja stron www

Mamy pewien system do kategoryzowanie stron internetowych. Zadanie polegałoby na zaimplementowaniu nowych algorytmów lub/i wykorzystaniu większej ilości informacji zawartych w dokumentach html, w celu uzyskania lepszych wyników. Zadanie jest o tyle ciekawe, że swoje pomysły można przetestować na bardzo dużej ilości polskich stron internetowych.

Realizacja zadania wymagałaby udostępnienia komputera studentowi z dostępem do danych pajączka w naszej firmie.

8. Operacje AIO - select/poll

- wersja na Linux
- wersja na FreeBSD

Tu znowu, albo biblioteka, albo moduł do kernela (chętniej to drugie - zarówno Linux jak i FreeBSD). System do odczytów i zapisów "w tle", ale zgłaszający fakt zakończenia operacji jako zdarzenie przerywające select'a/poll'a. Obecnie istnieją w Unix'ach operacje "aio". Niestety "event'y" związane z zakończeniem operacji I/O nie powodują przerwania funkcji typowych dla systemów asynchronicznej obsługi gniazd. Stąd
jak ktoś pisze typowy jednowątkowy serwer używający select'a/poll'a, czy też gotowych bibliotek takich jak libevent to niestety nie może używać równolegle systemu odczytów i zapisów "w tle". Do "aio" została
dodana jakaś niezależna funkcja informująca o zakończeniu jakiegoś odczytu, czy zapisu i nie da się równolegle czekać na zdarzenia z "aio" i inne dotyczące gniazd.

9. System do automatycznych tłumaczeń

Opracowanie postprocesora do php/apache umożliwiającego:
1. podmianę w wynikowej stronie identyfikatorów w określonej postaci na określone wartości (stringi).
2. opracowanie systemu do tłumaczeń wykorzystującego ww. Postprocesor oraz technologię ajax tak zmieniającego stronę, aby zamiast identyfikatora umieścić kod umożliwiający edycję wartości, która jest mu przypisana, oraz po zatwierdzeniu umieszczenie jej w bazie, przeładowanie wartości (ajax), i wyświetlenie nowej wartości (system tłumaczeń online).

10. System do testowania obciążenia serwera http

W idealnym przypadku system, który pozwala symulować równolegle zachowanie

kilkunastu tysięcy użytkowników, którzy zamęczają serwer HTTP. Najchętniej z różnych IP (nie wiem do końca jak to można zrobić - może trzebaby jakiś moduł do kernela w tym celu zmajstrować). System powinien móc stwierdzić np. jak szybko serwer odpowiadał (w zależności od ilości odwołań, czy zmieniało się to w czasie - może o równej godzinie serwer zamiera bo system bzip'uje logi ? - dobrze by było wykrywać takie rzeczy). Zakładam, że system by chodził przez jakiś czas i w wyniku podawał raport (może jakieś wykresiki?)

11. IOCTL do fuse

Dla niewtajemniczonych. Tworzenie własnych modułów obsługi filesystemów jest zadaniem niewdzięcznym. Głównie dlatego, że trzeba pisać moduły do jądra systemu operacyjnego - co jest dosyć trudne. Wymaga często specyficznej dla danego systemu operacyjnego wiedzy oraz jest praktycznie nieprzenośne pomiędzy różnymi systemami. Jakiś czas temu pojawiło się uniwersalne i przenośne rozwiązanie - FUSE (Filesystem in Userspace - fuse.sourceforge.net).

Jest to bardzo dobre rozwiązanie sprowadzające się do stosunkowo prostego modułu do jądra (obecnie dostępne dla systemów Linux, FreeBSD, Mac OS X i OpenSolaris) oraz biblioteki komunikującej się z tym modułem. Aby stworzyć własny FS wystarczy napisać proces typu 'daemon' używający odpowiednio tej biblioteki.

Obecnie FUSE wspiera większość operacji nowoczesnego systemu plików. Niestety brakuje w FUSE obsługi odwołań IOCTL, która to umożliwiłaby dodatkowe sterowanie obiektami systemu plików. Oczywiście dodanie ogólnie obsługi IOCTL nie jest możliwe, gdyż w ogólności nie jest znany rozmiar przekazywanej do IOCTL struktury. Nie mniej wydaje się, że można by na potrzeby FUSE'a dodać obsługę pewnego zakresu komend IOCTL z bardzo ogólną strukturą - typu "długość,dane". Tego typu podejście umożliwiłoby łatwe przekazywanie tych IOCTL'i (długość w prefiksie pozwoliłaby na to) do procesu daemona. Byłoby wskazane wprowadzenie tego w porozumieniu z autorem FUSE i najlepiej we wszystkich dostępnych systemach operacyjnych.

12. Interfejs offline do danych gemiusTraffic

Gemius moze generowac pliki xml zawierajace zestaw statystyk danego konta: Dla kazdego wezla: Odslony, wizyty, uzytkownicy, czasy itp. W podziale na godziny, dni, tygodnie i miesiace. Ostatnie wizyty, strony odsylajace, wyszukiwarki, geolokalizacja, itp. W zasadzie, w jednym pliku moze sie znalezc niemal wszystko
to, co można wyklikac w interfejsie gemiusTraffica.

Moznaby dopracowac format takiego pliku, a nastepnie wykonac offline'owa aplikacje, wczytujaca taki plik i prezentujaca dane. Nawet prezentacja taka sama, jak w obecnym interfejsie gemiusTraffic ma juz sens, bo mozna sobie zachowac taki plik i przegladac go gdzies, gdzie nie ma netu. Albo traktowac jako backup.

Jednoczesnie sa tu duze mozliwosci rozbudowy: Zaawansowane porownywanie statystyk roznych wezlow, szukanie wezlow z nietypowym zaburzeniem ruchu, prognozowanie, wyliczanie dodatkowych statystyk na podstawie innych danych (na przyklad zwrot kosztow kampanii reklamowych, po podaniu dat inwestycji), itp.

13. Poprawić SpamAssain

Powszechnie używany przez administratorów serwerów pocztowych SpamAssaim do filtrowania spamu jest napisany w perl-u. O ile chodzi o bazy danych, na jakich program bazuje, nie ma żadnych zastrzeżeń. Natomiast program jest mało efektywny. Być może stosując znane algorytmy typu Aho-Corasick (lub inne) – możnaby napisać w C lub C++, z wykorzystaniem wyższej algorytmiki, efektywniejszy program.

Temat, w zależności od stopnia zdefiniowania, mógłby zapewne być potraktowany jako temat pracy magisterskiej (a może doktorskiej?). Program byłby z wdzięcznością przyjęty przez administratorów serwerów pocztowych.

14. XMPortal

Celem projektu jest stworzenie wydajnego CMS'a, napisanego we frameworku Django, który potrafiłby zastąpić system XMPortal użyty do zbudowania strony niektórych wydziałów uniwersyteckich oraz ptm. System powinien wspierać technologie użytą we wspomnianym XMPortalu (xml+xslt), ale powinien wprowadzać również nowe typy węzłów (np. Html generowany przez szablony Django). Szablony stron powinny być obiektami w drzewie dokumentów (a nie jak obecnie funkcjonować poza nim).

Powinien być wbudowany system zarządzania prawami dla użytkowników, sami użytkownicy powinni jednak być definiowani poza tym systemem (w przypadku stron uniwersyteckich może to być system USOSweb i system centralnego logowania CAS). System nie powinien zawierać danych osobowych oraz nie powinno istnieć w nim konto centralnego administratora (administratorem powinien być jeden ze"zwykłych" użytkowników systemu).

Nowy CMS powinien również wspierać technologie AJAX, która rozsądnie używana pozwala na znaczne zmniejszenie ruchu. Powinny również istnieć rozbudowane formularze napisane w DHTML, które ułatwiałyby administrowanie stroną. Jako formaty wyjściowe powinny być obsługiwane wszelkie odmiany (x)html'a, xml'a oraz PDF. Jako formaty wejściowe powinny być obsługiwane formaty M$ Office i OpenOffice (jak również wspominane wcześniej formaty wyjściowe). Konwersja np. pomiędzy xsl a html powinna być zrealizowana jak np. w przypadku tych dwóch dokumentów: http://www.fuw.edu.pl/studia/rozklad/0809z/ii_i.xls i http://www.fuw.edu.pl/studia/rozklad/0809z/ii_i.xml. Powinien również powstać system generowania ogłoszeń i konwerter z ogłoszeń w obecnie istniejącym systemie.

W celu ułatwienia migracji ze starego systemu nowy CMS mógłby umożliwiać uruchamianie programów w php. W nowym systemie mogły by za to zniknąć profile - nigdy z nich nie korzystaliśmy.

15.Rozwiązywanie dużych i specyficznych układów równań liniowych

Rozwiązujemy w firmie pewien układ równań liniowych: Ax = b Problem z tym zadaniem jest taki, że macierz A ma wymiary około 20 000 x 100, a docelowo: 200 000 x 100. Taki układ równań (o ile nie jest sprzeczny) ma nieskończenie wiele rozwiązań. Spośród wszystkich rozwiązań należy znaleźć wektor x, który jest nieujemny oraz minimalizuje odchylenie standardowe czyli minimalizuje wyrażenie: suma((x_i - ex)^2)/n, gdzie ex = suma(x_i)/n. Naiwne zastosowanie algorytmów programowania kwadratowego niestety nie działa (np.: algorytm dla mniejszego zadania potrzebuje zaalokować pamięć rzędu: (20 000)^2 * sizeof(double) czyli ok 3GB ram-u). Rozwiązanie można oprzeć o pewne metody iteracyjne lub z użyciem

dysku lub wykorzystać fakt, że macierz złożona jest w większości z zer i jedynek. Do naszych celów najlepiej by było, aby rozwiązanie było napisane w C/C++.

Jeżeli się uda powyższe zadanie, można je rozszerzyć o fakt, że macierz A jest mimo wszystko często sprzeczna, i np.: próbować wyrzucać wiersze powodujące sprzeczność (jeżeli się nie mylę, to to zadanie jest NP-trudne, ale ponieważ wierszy jest niedużo to może ten fakt nie będzie stanowił problemu), lub szukać takiego x', żeby minimalizować błąd:
|| A * x' - y ||