TESTOWAĆ
TESTOWAĆ
TESTOWAĆ
TESTOWAĆ
TESTOWAĆ
TESTOWAĆ
TESTOWAĆ

Testowanie i naprawa Loxima

You cannot control what you cannot measure.

Tom DeMarco

Grzegorz Timoszuk

gt219709@students.mimuw.edu.pl

Plan prezentacji

- Trudne początki.
- Jak to poprawić?
- Co i jak mierzyć?
- Co i jak sprawdzamy:
  - wycieki pamięci,
  - alokacje pamięci,
  - problemy ze współbieżnością.
- Plany na przyszłość.

Trudne początki – opis Loxima

- Bardzo mało komentarzy.
- Jakość kodu:
  - nieprzemyślana struktura,
  - nieznajomość C++,
  - niechlujność,
  - i tak nikt nigdy tego nie otworzy...
- Nie wiadomo do końca co działa a co nie...
- **A JEDNAK DZIAŁA!!!**

Jak to poprawić?

- Od czego zacząć?
- Co poprawiać?
- Jak bardzo ingerować w kod?
- Jakich narzędzi używać? Jak je wybrać?
- Jak zabrać się do testowania?
- Dlaczego tak ważne są testy?
- Dlaczego ciężko napisać dobre testy?
- Dlaczego warto mieć środowisko do testów?

Miary

- Na razie 3 aspetky:
  - wycieki pamięci,
  - alokowana pamięć,
  - problemy ze współbieżnością.
- Propozycje?

- Wykonywane testy:
  - stukrotne połączenie,
  - 20 połączeń naraz,
  - 1 close by 1,
  - 12 kB insert,
  - 12 kB insert i select na nim,
  - 3x i 5x ten sam 12 kB insert.
- Propozycje?

Nieznajomośc C++

- Przekazywanie przez wartość a nie przez referencję.
- Template w pliku .cpp – jak to źle zrobić, żeby się kompilowało...
- Problemy z kompilacją.
- Tworzenie niepotrzebnych zmiennych na stercie
- .....

Wycieki pamięci

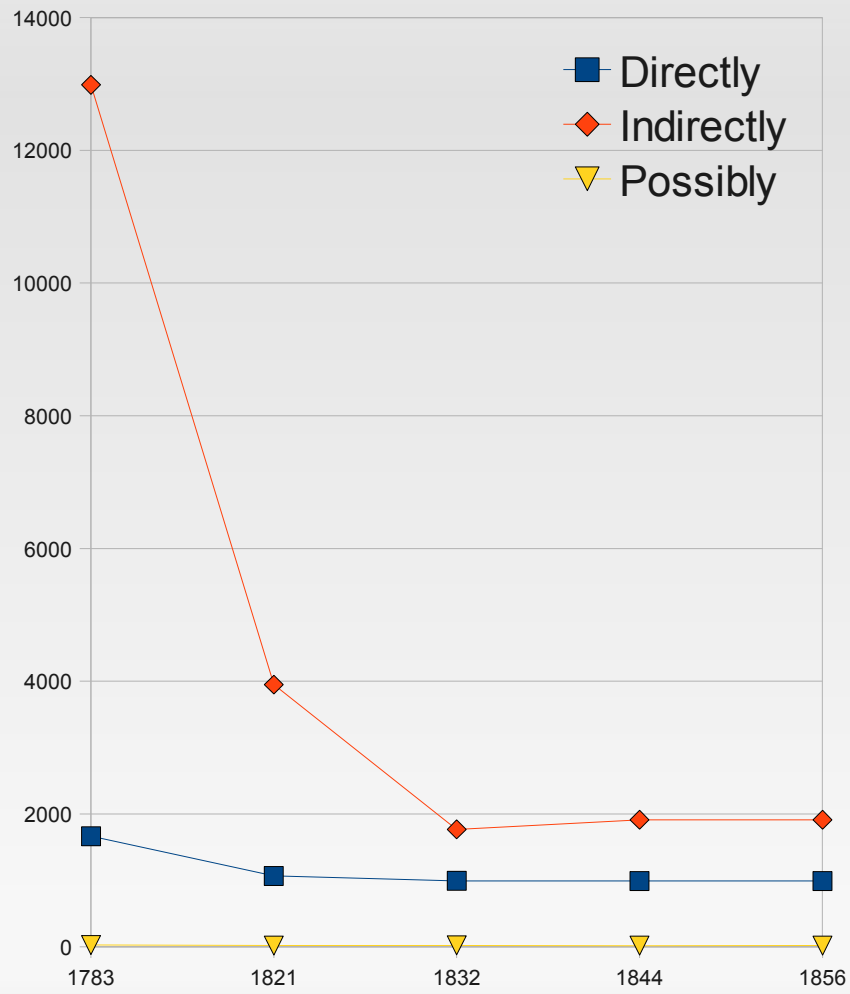
- Przyczyny.
- Zapobieganie:
  - RAII,
  - zliczanie referencji,
  - menadżer pamięci.
- Narzędzia:
  - inwazyjne: ccmalloc, mpatrol,
  - bezinwazyjne: YAMD, insure++, Valgrind.
- Wybór: Valgrind memcheck.

Valgrind memcheck

- Użuwają duże projekty – się sprawdza.
- Co można wykrywać:
  - Wycieki pamięci wg klasyfikacji
    - still reachable
    - possibly lost
    - definitely lost (direct, indirect)
  - Niepasujące do siebie alokacje/dealokacje.
  - Używanie niezainicjalozowanych danych.

Loxim, a wycieki pamięci

- Test 12 kB instert.
- Wycieki:
  - 60% to proste i duże
  - 20% to trudne i duże
  - 20% to małe i ciężko dokładnie określić ich status.
- Docelowo < 1MB.



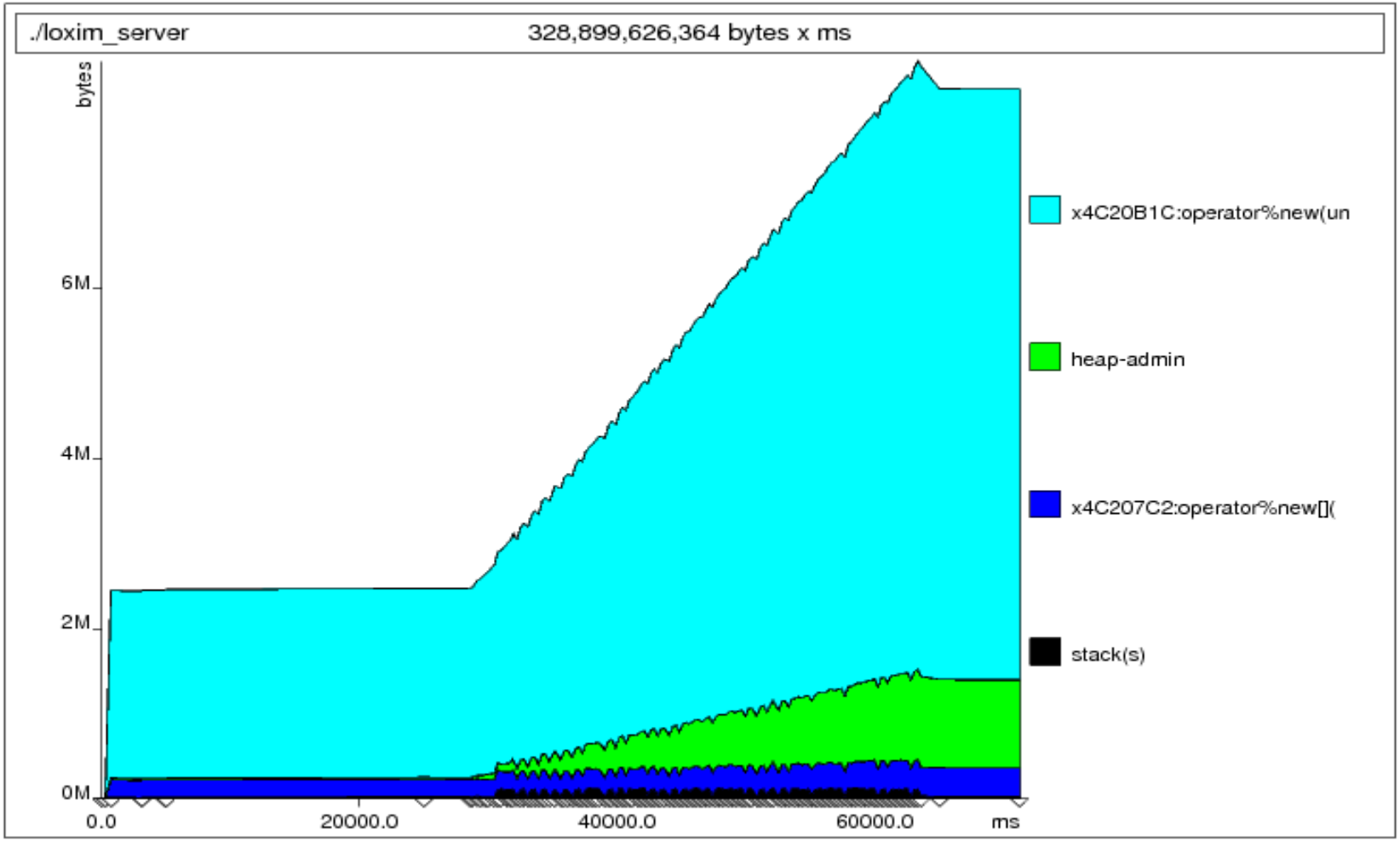
Optymalizacja sterty

- Jaka jest motywacja?
- Narzędzia:
  - Stary Valgrind Massif,
  - Nowy Valgrind Massif,
  - Google Performance Tools.
- Jak wygląda Loxim w tej metryce?

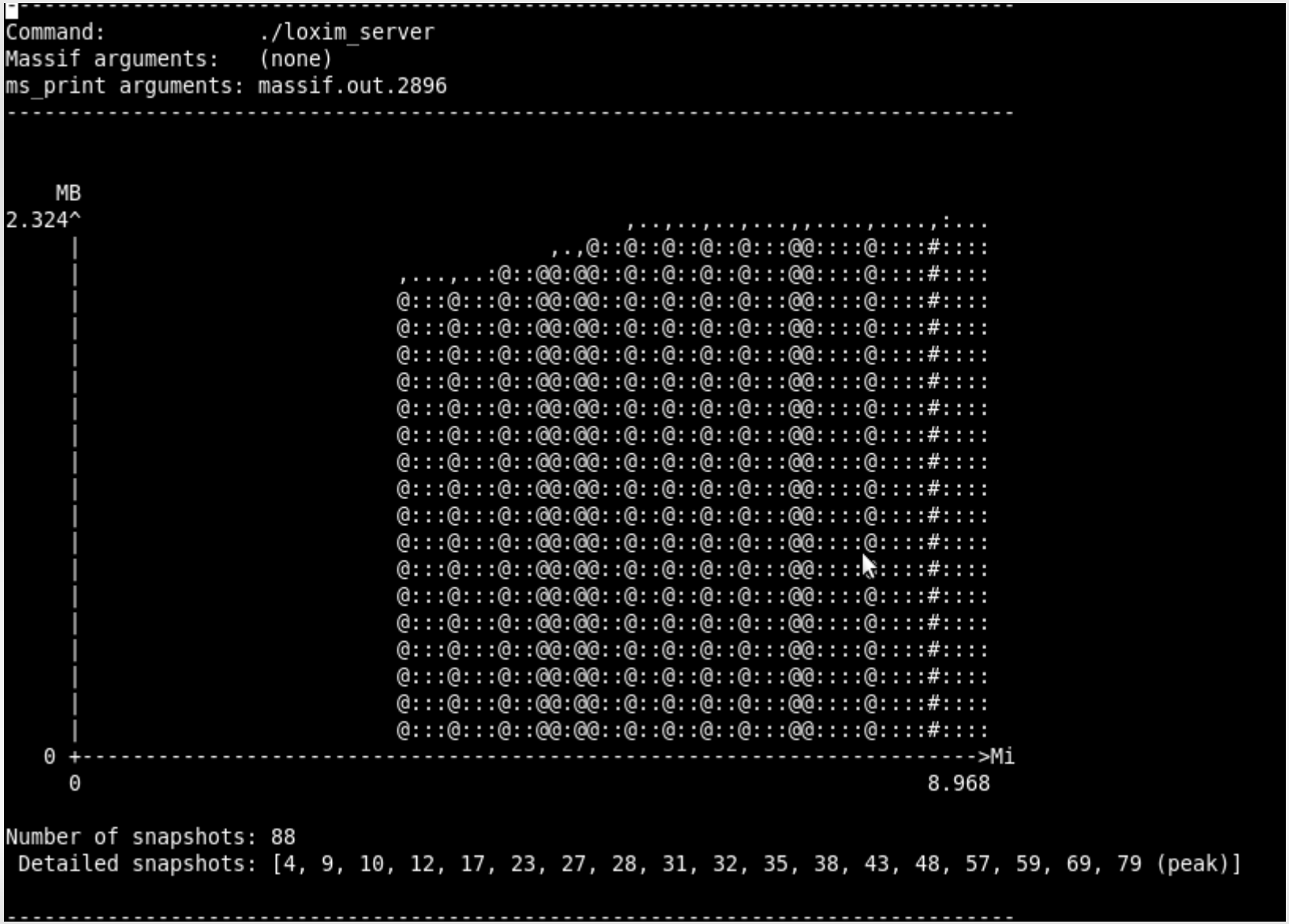
Valgrind Massif

- Pokazuje użycie sterty w czasie.
- Bezinwazyjny.
- Potrafi też śledzić zużycie stosu w czasie.
- Duża konfigurowalność.
- Bardzo dobry do długich złożonych testów.
- Dlaczego w chwili obecnej nie?
- Dlaczego kiedyś może tak...

Stary Massif...



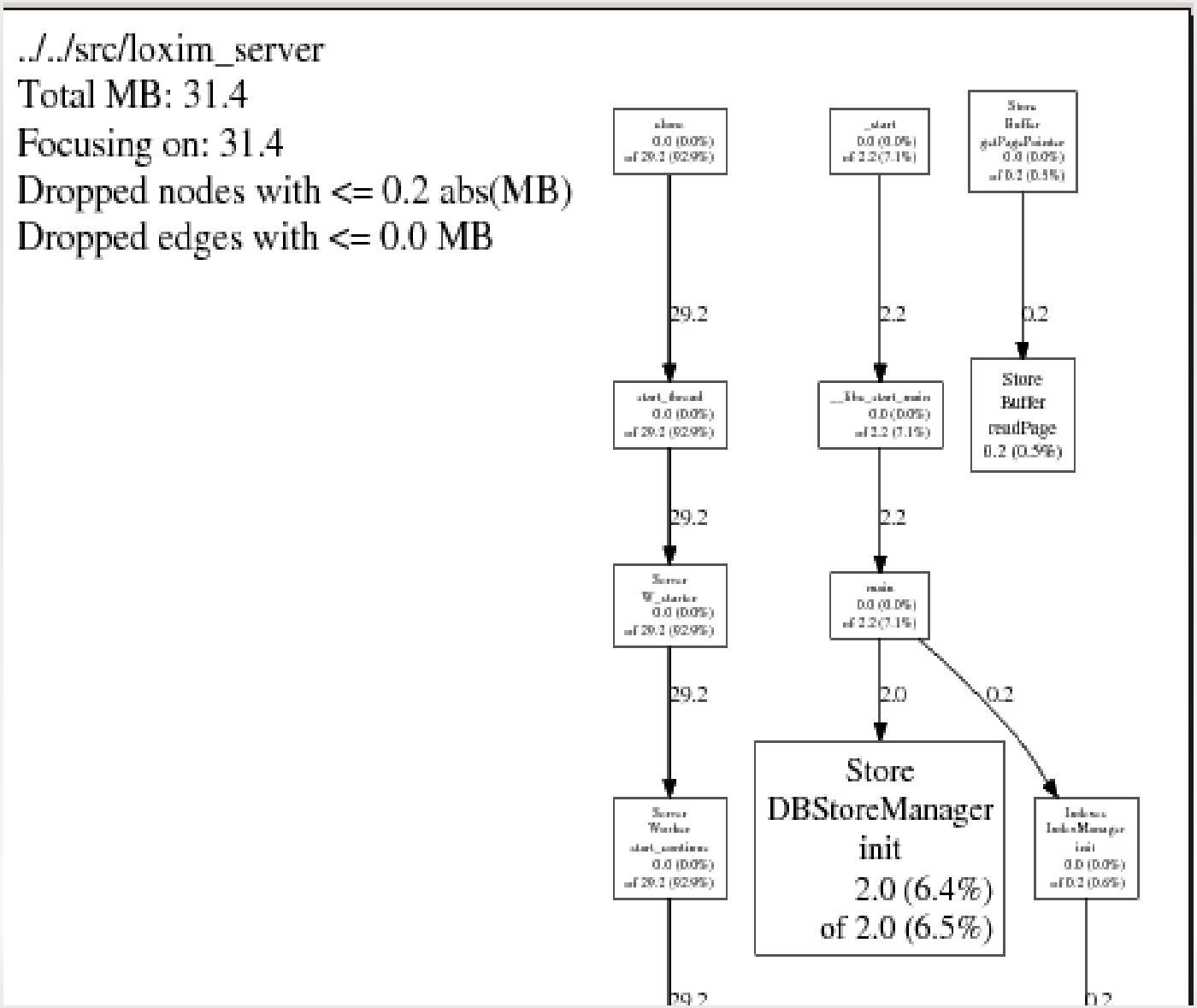
Nowy Massif...


Command: ./loxim_server
Massif arguments: (none)
ms_print arguments: massif.out.2896
MB
2.324^
0
8.968
Number of snapshots: 88
Detailed snapshots: [4, 9, 10, 12, 17, 23, 27, 28, 31, 32, 35, 38, 43, 48, 57, 59, 69, 79 (peak)]

Google Performance Tools

- Inwazyjny, trzeba się linkować ich bilioteką.
- Analizuje 4 aspekty:
  - inuse_space
  - inuse_objects
  - alloc_space
  - alloc_objects
- Może służyć też jako narzędzie do wykrwania wycieków pamięci.

Google Performance Tools



Problemy ze współbieżnością

- Dlaczego to tak ważne?
- Bardzo łatwo popsuć!!!
- Dlaczego tak trudno mierzyć?
- Dlaczego sprawdzanie przez czytanie kodu potrzebuje wspomagania.
- Tracimy jeden semafor na sejsę – a standardowo Linux ma ogracziczenia na 128...

Valgrind helgrind

- Wygodny czytelny output.
- Wybrany bo dobra marka...
- Aspekty, które są testowane:
  - Wyścigi,
  - Zakleszczenia,
  - Błędne użycie POSIX'owego interfejsu Pthreads.
- Tak naprawdę, nie taki wspaniały...
- Może jednak warto coś innego?

Przykłady innych narzędzi

- Google Performance Tools umie też wykrywać wycieki pamięci.
- Sun Studio.
- Valgrind Cachegrind i Valgrind Callgrind.
- Kcachegrind.
- Dużo innych narzędzi do mierzenia sterty...

Co jeszcze można mierzyć?

- Wywołania funkcji.
- Wykorzystanie cache.
- Czas wykonywania.
- Wytrzymałość na duże obciążenia.
- Stan po długotrwałej pracy.
- Ile stron musimy wczytać?

Plany na przyszłość

- Powstaje komplesowe środowisko do testów:
  - poprawnościowych,
  - wydajnościowych/jakościowych,
  - wprowadzenie testów długotrwałych..
- Zchodzimy z wyciekami pamięci poniżej ustalonego poziomu i ograniczamy pozostałe błędy związane z pamięcią.
- Rozwiązana chociaż część problemów związanych ze współbieżnością.

Pytania? Dyskusja!

Wesołych Świąt!!!.

- Grafika pochodzi z www.garfield.com