

# Rubyとキャッシュ

RejectKaigi 2007

よしおかひろたか  
ミラクル・リナックス株式会社  
カーネル読書会世話人  
(YLUG)

カーネル読書会世話人しています。

99年4月～

次回は7.6回目

仕事

ミラクル・リナックス  
CTO

開発者募集中～

<http://blog.miraclelinux.com/yume/>

# Ruby歴

実質3日くらいか？

素人とはわたしの事だ

Rubyに対する貢献

会社(Asianux)のポロ  
シャツを

# Matzにあげた



Asianuxポロシャツ  
を着るMatz

Matzいわく

これからは機能ではなく  
実装を考えたい。  
日本Ruby会議2007

実装と言えは

# 性能命 ムーアの法則

## ムーアの法則

半導体の集積度は  
18ヶ月で倍になる～

集積度が倍になったら。  
ら。

集積度が倍になったら

メモリが増える

クロックが速くなる

↑いまここ

マルチコア  
↑これから

# なぜキャッシュが重要か？

CPUの速度向上（年率50%以上）に比べてメモリアクセスの向上率は低い（年率7%程度）

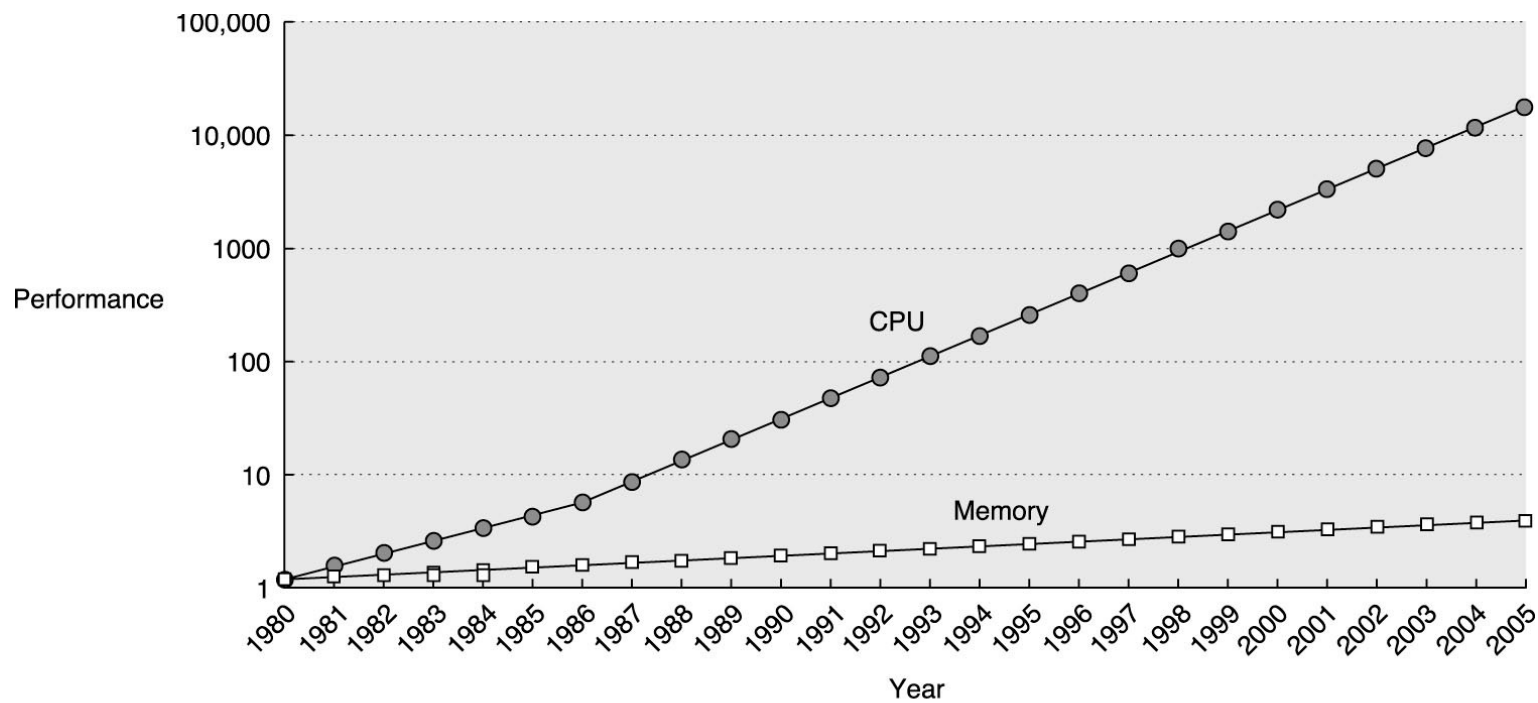
メモリアクセスのペナルティは増加傾向

L1 2clock(1ns)

L2 10clock(5ns)

メモリ 200~300clock以上  
(150ns)

# CPUとメモリ速度向上



© 2003 Elsevier Science (USA). All rights reserved.

# なぜRubyをハックするのか 動機重要～

なぜRubyをハックするのか

彼女ができたらしい  
> ささださん

なぜRubyをハックするのか

モテたいから

↑ 重要

ということで

Rubyをハックすると、  
もてるかどうか検証し  
てみた。

# Rubyをハックする

キヤッシュミスを減  
らす

とこのをゴールに  
してみた。(もてた  
いから)

Rubyのキャッシュミスが減らす

キャッシュミスを測  
定する

キャッシュミスのボ  
トルネットワークを分析す  
る

ハックする

準備するもの

Rubyのソース

Linuxが動くマシン

ベンチマークプログラム

oprofile

キャッシュミスを測定する

どこでキャッシュミス?

プロファイルをとる  
すげー簡単

プロファイルのとり方。

```
# opcontrol --start;  
ruby runner.rb;  
opcontrol --stop
```

(一行で書いてね)

opreport -dg 詳細が下記のように表示される。

CPU: Core Solo / Duo, speed 2666.77 MHz  
(estimated)

Counted DCACHE\_PEND\_MISS events (Weighted  
cycles of L1 miss outstanding) with  
a unit mask of 0x00 (Weighted cycles) count 100000

vma	samples	%	linenr	info	app
00000000000042be50	244787	33.2383		gc.c:1661	
	ruby	os_each_obj			
00000000000042bfac	1	4.1e-04		gc.c:1599	
00000000000042bfb9	5	0.0020		gc.c:1599	
00000000000042bfbe	6	0.0025		gc.c:1599	
00000000000042bfd0	4862	1.9862		gc.c:1601	
00000000000042bfd3	228573	93.3763		gc.c:1601	
00000000000042bfd6	2698	1.1022		gc.c:1601	
00000000000042bfd8	250	0.1021		ruby.h:672	

```

static VALUE
os_obj_of(of)
  VALUE of;
{
  int i;
  int n = 0;

  for (i = 0; i < heaps_used; i++) {
    RVALUE *p, *pend;

    p = heaps[i].slot; pend = p + heaps[i].limit;
    for (; p < pend; p++) {
      if (p->as.basic.flags) { /* この部分 */
        switch (TYPE(p)) {
          case T_ICLASS:
          case T_VARMAP:
          case T_SCOPE:
          case T_NODE:
            continue;
          case T_CLASS:
            if (FL_TEST(p, FL_SINGLETON)) continue;
          default:
            if (!p->as.basic.klass) continue;
            if (rb_obj_is_kind_of((VALUE)p, of)) {
              rb_yield((VALUE)p);
              n++;
            }
        }
      }
    }
  }

  return INT2FIX(n);
}

```

# これがパッチだよ

```
-   for (i = 0; i < heaps_used; i++) {
+   RVALUE *p, *pend;
+   RVALUE *p, *pend;

    p = heaps[i].slot; pend = p +
heaps[i].limit;
    for (; p < pend; p++) {
+   if ( (p+1) < pend) {
+   asm __volatile__ (
+   " prefetch (%0)¥n"
+   :: "r" ((p+1)->as.basic.flags) );
+   }
    if (p->as.basic.flags) {
        switch (TYPE(p)) {
            case T_ICLASS:
```

L1 キャッシュミスが  
41%減った (拍手)

ここまで約3時間 (す  
ごい)

Rubyをハックするともてるか

orz

カーネル読書会参加  
者絶賛募集中

# Happy Hacking