

Génération de code binaire pour application multimedia : une approche au vol

<http://hpbcg.org/>

Henri-Pierre Charles

Université de Versailles Saint-Quentin en Yvelines

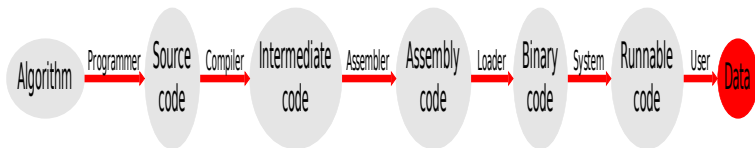
3 Octobre 2009

Présentation

- Université de Versailles
- Laboratoire PRiSM
- Optimisation de code
- Partenariats académiques industriels : CEA, Intel, Bull

Le défaut de performance est devenu un bug

Chaîne de compilation



Code source est supposé “haut niveau”

Code exécutable est supposé “bas niveau”

Optimisation est supposée “indépendantes des données”

La machine cible est supposée “connue”

Plus rien n'est vrai !

Ce que l'on demande à un compilateur

1980 produire un code correct

2009

- ① produire un code rapide
- ② parallélisation automatique (multi CPU / multi core / multi thread)
- ③ vectoriser le code
- ④ utiliser les jeux d'instruction multimedia
- ⑤ instrumenter le code

Ask for program !

Quelles sont les variations de performance de ce programme ?

```
int i;
for (i= 0;  i < N;  ++i)
{
    int j;
    dest[i]= 0;
    for (j= 0;  j < N;  ++j)
        dest[i] += src[j] * m[i][j];
}
```

Variables : choix du compilateur, taille des données, processeur cible, parallélisme disponible, type de données, emplacement mémoire, système d'exploitation, ...

La taille compte (. . . La taille des données, a quoi pensiez vous ?)

Taille de la boucle (valeur de N)

10^1 Multimedia kernel : Full loop unroll, instruction scheduling, memory caches access, ...

$10^2/10^3/$ Scientific code : loop unroll, loop conversion, data prefetching

10^6 Multimedia flux : multithreading

10^{10} *andmore* High level parallelism : MPI / Grid / Cloud, ...

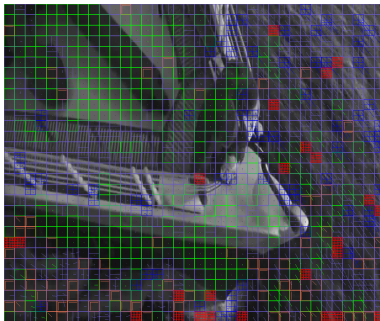
N is generally a parameter only known at run-time. Profiling and Iterative compilation does not help.

Compilation strategies are complex and are application domain specific

“Titanic” effet

The "Titanic" effect 1/2

Data set modify performance application during run-time

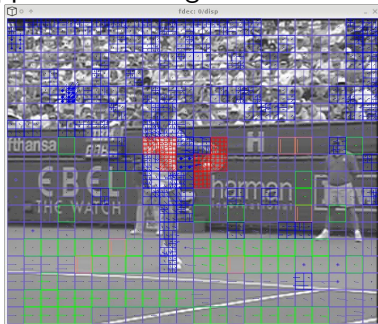
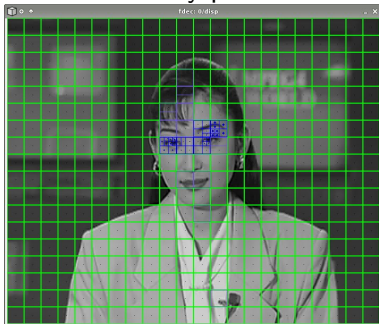


Using a FreeBOX (ADSL Modem and TV/IP broadcasting), my home computer (Intel Celeron 1.70GHz) play correctly France3 channel (which use MPEG TS video format) but cannot play RTL9 (X264 video format)

Optimization is not data independant

The “Titanic” effect 2/2

Data set modify performance application during run-time



Extract from MPEG group benchmarks

Optimization is not data independant

C'est compliqué un compilateur au vol ? NON !

C'est compliqué un compilateur au vol ? NON !

```
unsigned char code[] =
{
    0x78, 0x81, 0x01, 0x83,      /* mpy $3,$3,$4 */
    0x35, 0x00, 0x00, 0x00      /* bi  $0 */
};
../..
typedef int (*pfiii)(int, int);
pfiii Sad = (pfiii) code;
int result = Sad(2, 21);
```

Write a "complette" (thinking time)

Write a "complette" (thinking time)

```
pifi multiplyCompile(int multiplyValue)
{
    insn *code= (insn *)_malloc_align(1024, 7);
    (void) printf("Code generation for multiply value %d\n", mu
    #[
        .org      code
        mpyi      $3, $3, (multiplyValue)
        bi $lr
    ]#;
    (void) printf("Code generated\n");
    return (pifi)code;
}
```

Write a "complette" (thinking time)

Use the "complette"

```
/* Generate binary code */
multiplyFunc = multiplyCompile(atoi(argv[1]));
for (i = 1; i < 11; ++i)
    printf("%3d ", i);
printf("\n");
for (i = 1; i < 11; ++i)
    printf("%3d ", multiplyFunc(i));
printf("\n");
```

Write a "complette" (thinking time)

Generate the code generator (Static compile time)

```
hpbcg simple-multiply-cell.hg > simple-multiply-cell.c
cc -Wall ../.. simple-multiply-cell.c -o simple-multiply-cell.c
spu-gcc ../.. -o simple-worker-cell simple-worker-cell.c
```

Write a "complette" (thinking time)

Run the code generator (Run time)

```
turner:simple-multiply/>./simple-multiply-cell 6
```

```
Code generation for multiply value 6
```

```
Code generated
```

1	2	3	4	5	6	7	8	9	10
6	12	18	24	30	36	42	48	54	60

```
turner:simple-multiply/>./simple-multiply-cell 7
```

```
Code generation for multiply value 7
```

```
Code generated
```

1	2	3	4	5	6	7	8	9	10
7	14	21	28	35	42	49	56	63	70

```
turner:simple-multiply/>./simple-multiply-cell 14
```

```
Code generation for multiply value 14
```

```
Code generated
```

1	2	3	4	5	6	7	8	9	10	
---	---	---	---	---	---	---	---	---	----	--

Write a "complette" (thinking time)

Complettes / Results

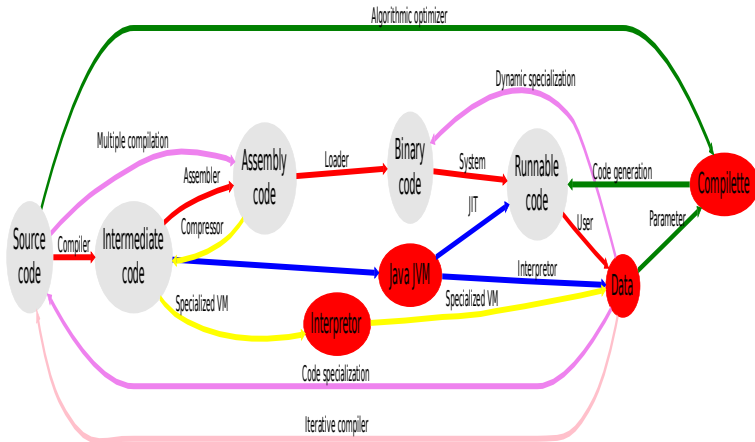
Obtained goals :

- 1 Matrix vector vectorisation
- 2 Matrix vector optimization
- 3 Image filtering optimisation (using multimedia instructions)
- 4 X264 video encoder optimization (data alignment, multimedia instructions)
- 5 OpenGL off screen optimization (vectorization, multimedia instructions, optimization)

Multi target : Sparc Itanium, Power (Power & AltiVec), Cell (SPU & PPU),
CCG / HPBCG

General goal : experiment new compilation process

General goal : experiment new compilation process



TODO list

- Go high level
 - Find a “complette” generator (gcc-LTO project, C++, python, ...)
 - Use it in the large (STAPL)
- Find a new position !

Ask for the program

- <http://hpcbg.org/>
- Support for :
 - **power** power4, altivec, cell, FP2
 - **itanium** full isa but not scheduling
 - **arm** preliminary
- Samples codes
- BSD like licence