Hancock

Ζωγραφάκης Ιωάννης X346339
Εξαρχάκος Νικόλαος T911778

ΕΠΛ 428 – Προγραμματισμός Συστημάτων

Ιστορική Αναδρομή

- Δημιουργήθηκε από την εταιρεία AT&T LAB
- Αφορμή δημιουργίας: Η ανάγκη για καθαρό και αποδοτικό κώδικα για transaction streams

Τι είναι το stream;

- Ιστορικό, το οποίο μας δείχνει την αλληλεπίδραση μεταξύ δύο οντοτήτων
- Αποτελεί τον κύριο όγκο πληροφοριών τον οποίο επεξεργάζεται η γλώσσα

Πλεονεκτήματα

- Μπορεί να διαχειριστεί μεγάλο όγκο ομοιόμορφων πληροφοριών
- Είναι C based language
- Είναι scripting language
- Μειώνει τον όγκο πληροφοριών
- Εύκολη στην εκμάθηση
- Μικρότερος κώδικας
- Δε μας ενδιαφέρει ο όγκος των πληροφοριών που δεχόμαστε

Μειονεκτήματα

- Ελάχιστος όγκος πληροφοριών για την γλώσσα
- Δεν υπάρχουν εργαλεία για την γλώσσα
- Αργοί υπολογισμοί άρα και σχετικά αργή γλώσσα

Εγκατάσταση

- http://www.research.att.com/~kfisher/hancock/release.php
- Υποστηρίζει συστήματα Solaris, MacOs, Linux.

HELLO WORLD

- Είναι C embedded language

  printf(“ hello world \n”);

$hello world

Μεταγλώττιση

- Παρόμοιος τρόπος με μεταγλώτισση προγραμμάτων σε C
- hcc -o my.out my.hc
- Hcc - - help

Μεταγλώττιση(flags)

-s Redirect output to the standard output stdout. This option can only be used in conjunction with the -E or -C flags.
-o Name the output le based on the string following -o. This flag can be used to name the output C le or resulting executable depending on what other flags are present. The default output filename depends on the type of the output file. For executables the default is a.out. For C files the default is the input filename with its extension replaced by .c.
-g Compile all code with debugging information. Link in the debugging version of the Hancock runtime.
-E Only pre-process the input files.
-C Compile the first file with extension .hc to a C le and then stop. Compiling to C involves pre-processing the file.
-I Add a directory to search for include les to the call to the pre-processor.
-D Add a denition to the call to the pre-processor.
-m Limit the number of error messages the compiler will generate before giving up.
-t Print a trace of all system commands executed by the compiler.

Hello Hancock

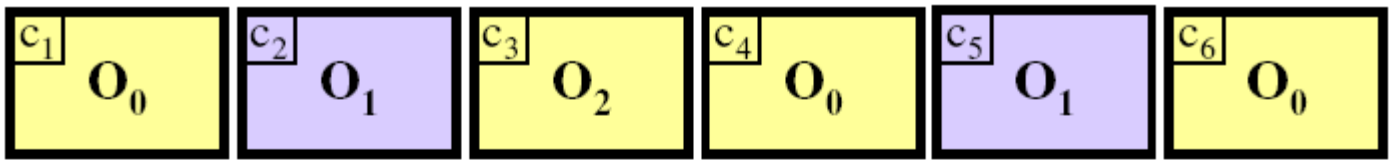
Παράδειγμα μέτρησης διεθνών κλήσεων

```
typedef struct{
    pn_t origin;
    pn_t dialed;
    .......
   char isIntl;
}scampRec_t;
```

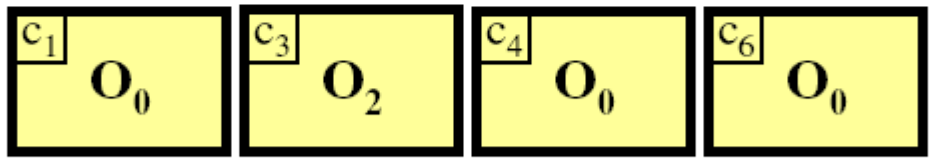
Hello Hancock

```
#include "scampRec.hh"                                          /* ( 1) */
#include <stdio.h>                                              /* ( 2) */
                                                                /* ( 3) */
void sig_main(callDetail_s callStream <c:>)                     /* ( 4) */
{                                                               /* ( 5) */
int msgs = 0;                                                   /* ( 6) */
                                                                /* ( 7) */
iterate                                                         /* ( 8) */
( over callStream                                               /* ( 9) */
filteredby(c) (c->isIntl) ) {                                   /* (10) */
                                                                /* (11) */
event (scampRec_t *c) {                                         /* (12) */
msgs++;     /* saw an international call */                     /* (13) */
}                                                               /* (14) */
};                                                              /* (15) */
                                                                /* (16) */
printf("There were %d international calls.\n", msgs);          /* (17) */
}                                                               /* (18) */
```

Hello Hancock



(a) Sample stream of calls.

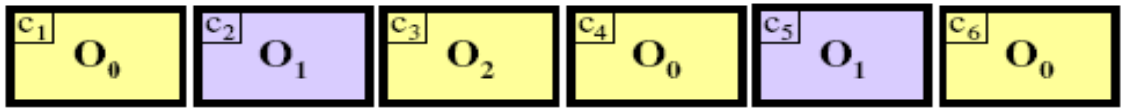


(b) After filtering.

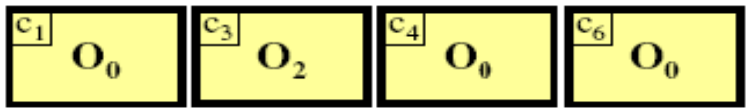
Hello Hancock

```
void sig_main(callDetail_s callStream <c:>,
intlCount_m ic <I:>)
{
int msgs;
iterate (
 over callStream
sortedby origin
filteredby(c) (c->isIntl)
withevents originDetect) {
        event line_begin(pn_t pn) {
        msgs = 0;
}
/* saw an international call
        event call(scampRec_t r) {
        msgs++;
}
/* update the count for pn
        event line_end(pn_t pn) {
        ic<:pn:> = msgs + ic<:pn:>;
 } };}
```

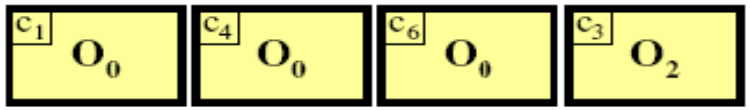
Hello Hancock



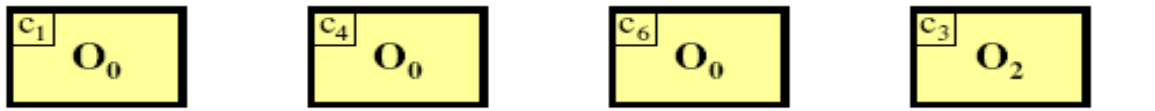
(a) Sample stream of calls.



(b) After filtering.



(c) After filtering and sorting.



begin_line(O_0)
call(c_1)

call(c_4)

call(c_6)
end_line(O_0)

begin_line(O_2)
call(c_3)
end_line(O_2)

Παραδείγματα Hancock

Το hancock υποστηρίζει 3 κύριους τύπους δεδομένων:

- Directories
- Maps
- Pickles

Directories

Μέσω των directories δημιουργούμε μια δομή που μπορεί περιέχει όλες τις υπόλοιπες

```
typedef struct {
    int x;
    float y;
} foo_t;

directory dir_d {
    int myInt default 0;
    float myFloat default 0.0;
    foo_t myStruct default {0, 0.0};
    int myArray[2] default {0,0};
    usage_m myMap;
};
```

Maps

- **Ένας τρόπος να συσχετίσουμε πληροφορίες με μια πληροφορία-κλειδί**
- **map** usage_m {

  key (MINVALIDPN .. MAXVALIDPN-1);

  split (10000, 100);

  value uApprox_t;

  default {0, 0, 0};

};

Pickles

- **Δομή δεδομένων που χρησιμεύει σαν ένα είδος buffer**
- **pickle** pht_p {init_pht => pht_rep => flush_pht};
- int init_pht(Sfio_t *fp, pht_rep *data, char readonly);
- int flush_pht(Sfio_t *fp, pht_rep *data, char close);

Συμπεράσματα

- Χρήσιμη γλώσσα
- Εύκολη στη γραφή, ανάγνωση και συντήρηση
- Μπορεί να δεχτεί μεγάλο όγκο πληροφοριών, με τίμημα το χρόνο εκτέλεσης
- Ικανοποιητικά αποτελέσματα

Βιβλιογραφία

- http://www.research.att.com/~kfisher/hancock/manual.pdf
- http://www.researchchannel.org/prog/displayevent.aspx?rID=2113&fID=569
- http://www.research.att.com/~kfisher/hancock/journal.pdf
- http://blog.wired.com/27bstroke6/2007/10/att-invents-pro.html

ΤΕΛΟΣ

- Απορίες??
- Χειροκρότημα?? ☺