Institut Supérieur d'Électronique de Paris
Projet de Fin d'Études

Reponsable: M. Hugueney

Finite State Transducers Just-In-Time Compiling

Do you hear the bytecode ?

Émilien Boulben
Victor Delepine
Corentin Peuvrel

17 juin 2015
Paris

Table des matières

Listings

Liste des tableaux

Table des figures

Introduction

1 Analyse préalable

2 Des FST en JIT

3 Optimiser : comment ?

Conclusion

A Annexe : tests avec un script shell

A.1 Dictionnaires

Value	Word
0	mop
1	moth
2	pop
3	star
4	stop
5	top

Tableau 1 – Dictionnaire à utiliser avec la FST dans le Tableau 3

Value	Word
0	mop
1	moth
2	pop
3	slop
4	sloth
5	stop
6	top

Tableau 2 – Dictionnaire à utiliser avec la FST dans le Tableau 4

A.2 FST

Nœu	0	0	0	0	1	2	3	2	4	6	7	5	7	8	
Nœu suivant	1	4	4	6	2	3	9	9	5	7	5	9	8	9	
Nœu final															9
Caractère	M	P	T	S	O	T	H	P	O	T	O	P	A	R	
Poids		2	5	3			1				1				

TABLEAU 3 – FST utilisée avec le dictionnaire Tableau 1, voir Figure 1 page 7

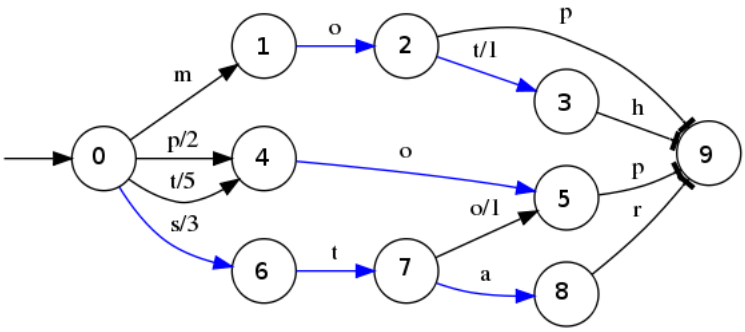

FIGURE 1 – La FST associée avec le Tableau 3 page 7

Nœu	0	0	0	0	3	3	1	2	4	5	6	5	
Nœu suivant	1	1	3	4	1	4	2	7	5	6	7	7	
Nœu final													7
Caractère	P	T	S	M	T	L	O	P	O	T	H	P	
Poids	2	6	3		2					1			

TABLEAU 4 – FST utilisée avec le dictionnaire Tableau 2, voir Figure 2 page 7

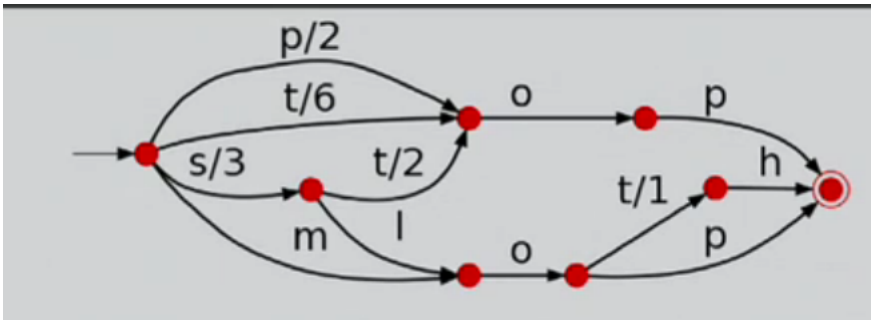

FIGURE 2 – La FST associée avec le Tableau 4 page 7

A.3 Générer à la volée du code C

A.3.1 Script shell

```
#!/bin/bash

FST="$1"

sort "$FST" > "$FST.sort"

FIRST_CALL=1

cat <<EOF
#include <stdio.h>

int compute_fst(const char* token)
{
    int pos=0;
    int total=0;

EOF

while read DEP ARR CHAR WEIGHT ; do
    # If it's a final node
    if [[ ! "$CHAR" ]]; then
        WEIGHT=${ARR:-0}
        cat <<EOF
    default:
        return -1;
    }

NODE_$DEP :
EOF
    (( WEIGHT != 0 )) &&
        echo "    total += $WEIGHT;"
    echo "    goto END;"

        continue
    fi

    : ${WEIGHT:=0}

    if [[ $DEP != $PREV_DEP ]]; then
        if [[ ! "$FIRST_CALL" ]]; then
            cat <<EOF
    default:
        return -1;
    }

EOF
        fi

        cat <<EOF
NODE_$DEP :
    pos++;
    switch (token[pos-1]) {
EOF
    fi

    echo "    case '$CHAR':"
```

```
57      (( WEIGHT != 0 )) &&
58          echo "          total += $WEIGHT;"
59      echo "          goto NODE_$ARR;"
60
61      PREV_DEP=$DEP
62      FIRST_CALL=
63  done < "$FST.sort"
64
65  cat <<EOF
66
67  END :
68      return total;
69  }
70
71  int main(int argc, const char *argv[])
72  {
73      if (argc < 2)
74          return 1;
75
76      printf("%d\n", compute_fst(argv[1]));
77      return 0;
78  }
79  EOF
80
81  rm "$FST.sort"
```

Listing 1 – Script pour générer un code C à la volée d'une FST

A.3.2 Code C généré pour la FST définie dans le Tableau 3

```c
#include <stdio.h>

int compute_fst(const char* token)
{
    int pos=0;
    int total=0;

NODE_0 :
    pos++;
    switch (token[pos-1]) {
    case 'M':
        goto NODE_1;
    case 'P':
        total += 2;
        goto NODE_4;
    case 'T':
        total += 5;
        goto NODE_4;
    case 'S':
        total += 3;
        goto NODE_6;
    default:
        return -1;
    }

NODE_1 :
    pos++;
    switch (token[pos-1]) {
    case 'O':
        goto NODE_2;
    default:
        return -1;
    }

NODE_2 :
    pos++;
    switch (token[pos-1]) {
    case 'T':
        total += 1;
        goto NODE_3;
    case 'P':
        goto NODE_9;
    default:
        return -1;
    }

NODE_3 :
    pos++;
    switch (token[pos-1]) {
    case 'H':
        goto NODE_9;
    default:
        return -1;
    }

NODE_4 :
    pos++;
    switch (token[pos-1]) {
```

```
59      case 'O':
60          goto NODE_5;
61      default:
62          return -1;
63      }
64
65  NODE_5 :
66      pos++;
67      switch (token[pos-1]) {
68      case 'P':
69          goto NODE_9;
70      default:
71          return -1;
72      }
73
74  NODE_6 :
75      pos++;
76      switch (token[pos-1]) {
77      case 'T':
78          goto NODE_7;
79      default:
80          return -1;
81      }
82
83  NODE_7 :
84      pos++;
85      switch (token[pos-1]) {
86      case 'O':
87          total += 1;
88          goto NODE_5;
89      case 'A':
90          goto NODE_8;
91      default:
92          return -1;
93      }
94
95  NODE_8 :
96      pos++;
97      switch (token[pos-1]) {
98      case 'R':
99          goto NODE_9;
100     default:
101         return -1;
102     }
103
104 NODE_9 :
105     goto END;
106
107 END :
108     return total;
109 }
110
111 int main(int argc, const char *argv[])
112 {
113     if (argc < 2)
114         return 1;
115
116     printf("%d\n", compute_fst(argv[1]));
117     return 0;
```

```
118 || }
```

Listing 2 – Code C généré pour la FST définie dans Tableau 3

A.3.3 Code C généré pour la FST définie dans le Tableau 4

```c
#include <stdio.h>

int compute_fst(const char* token)
{
    int pos=0;
    int total=0;

NODE_0 :
    pos++;
    switch (token[pos-1]) {
    case 'P':
        total += 2;
        goto NODE_1;
    case 'T':
        total += 6;
        goto NODE_1;
    case 'S':
        total += 3;
        goto NODE_3;
    case 'M':
        goto NODE_4;
    default:
        return -1;
    }

NODE_1 :
    pos++;
    switch (token[pos-1]) {
    case 'O':
        goto NODE_2;
    default:
        return -1;
    }

NODE_2 :
    pos++;
    switch (token[pos-1]) {
    case 'P':
        goto NODE_7;
    default:
        return -1;
    }

NODE_3 :
    pos++;
    switch (token[pos-1]) {
    case 'T':
        total += 2;
        goto NODE_1;
    case 'L':
        goto NODE_4;
    default:
        return -1;
    }

NODE_4 :
    pos++;
    switch (token[pos-1]) {
```

```
59      case 'O':
60          goto NODE_5;
61      default:
62          return -1;
63      }
64
65  NODE_5 :
66      pos++;
67      switch (token[pos-1]) {
68      case 'T':
69          total += 1;
70          goto NODE_6;
71      case 'P':
72          goto NODE_7;
73      default:
74          return -1;
75      }
76
77  NODE_6 :
78      pos++;
79      switch (token[pos-1]) {
80      case 'H':
81          goto NODE_7;
82      default:
83          return -1;
84      }
85
86  NODE_7 :
87      goto END;
88
89  END :
90      return total;
91  }
92
93  int main(int argc, const char *argv[])
94  {
95      if (argc < 2)
96          return 1;
97
98      printf("%d\n", compute_fst(argv[1]));
99      return 0;
100 }
```

Listing 3 – Code C généré pour la FST définie dans Tableau 4