

4.1 Le Translateur C vers Promela (CProl)

Le translateur est réalisé suivant le principe de translation adopté en chapitre 3, les outils et les langages utilisés sont les suivants :

4.1.1 Les Outils de développement

La Plateforme de développement .NET

La plate-forme .NET est un ensemble formé d'outils (présents dans le SDK), de modules actifs au runtime (le CLR par exemple) et de classes formant une API très étendue pour les applications de bureau comme pour les applications Internet. Des langages exploitent cette nouvelle base comme C# ou VB.NET.

Le Langage de developpement C# :

C# est un langage orienté objet élégant et de type sécurisé qui permet aux développeurs de générer une large gamme d'applications sécurisées et fiables qui s'exécutent sur le .NET Framework.

Le langage C et Promela:

Est le langage input des programmes sujet de la translation vers le langage cible Promela.

SPIN :

Est un outil général pour la vérification de la correction des logiciels distribués (la conception des logiciels) d'une manière rigoureuse et presque totalement automatique. Il prend le code Promela comme un module input pour la vérification. On utilise Spin pour vérifier les modèles Promela traduits et donc, les programmes C seront les modèles vérifiés.

Code::Blocks :

Est un (IDE) environnement de développement intégré libre et multiplate-forme. Il est écrit en C++ grâce à la bibliothèque wxWidgets. Pour le moment, Code::Blocks est orienté C et C++, mais il supporte d'autres langages.

4.2 La modélisation du translateur CProI

Notre translateur est conçu avec une vision objet. Cinq classes sont définies pour atteindre les objectifs perçues. Les classes ainsi que leur relation de composition sont représentées :

4.2.1 Diagramme de Classes

Figure 4.1 : Diagramme de classes

Figure 4.1 : Diagramme de classes

4.2.2 Description des Classes

La classe Token : décrit les propriétés pertinentes(kind, line, position, string-C, valeur-Promela) des tokens (tous les mots du programmes C).

La classe Buffer : modélise le code C-source et définit des Méthode pour parcourir et lire les caractères du code source.

La classe Scanner: Instancie un objet buffer au début une seule fois pour contenir tout le text de Code Source-C.

Instancie un objet token pour contenir le token détecté.
détecte les tokens et définit ses propriétés.

La classe Tools : contient le texte target Promela résultat de la traduction en cours.

Des méthodes assurent l'écriture du code Promela approprié (eg :convertir les types de Variables).

La classe Parser : implémente le principe de translation proposé.

Instancie tools, token et scanner.

4.2.3 L'interface principale :

Menu textuel standard:

- Fichier,
- Edition,
- aide

Barre d'outils graphiques:

- Ouvrir fichier.
- Convertir : lance la translation du C vers Promela
- Spin : lance le spin.
- Code::Blocks : IDE pour l'édition et la compilateur du code C.

Editeur de code C

Chapitre 4 : Traducteur C vers Promela

Afficheur de code Promela.

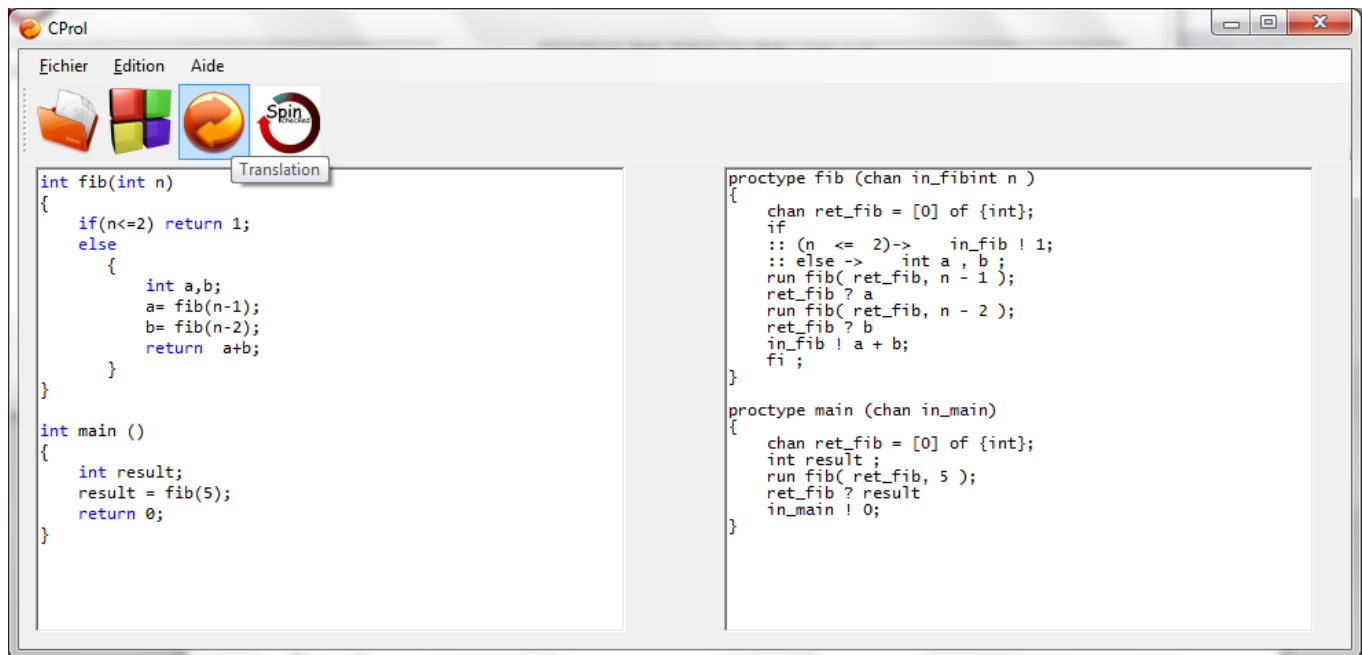


Figure 4.2 : L'interface de Traducteur CProL.



Figure 4.3 : L'interface de SPIN model Checker.

4.3 Exemples

ExempleI : programme de factorielle en c en utilisant la récursivité

Code C

```
int factorial(int n)
{
    if(n==0)
        return 1;
    else
    {
        int result ;
        result = factorial(n-1);

        return n*result;
    }
}
void main()
{
    int num;
    int f;

    f = factorial(num);

}
```

Promela

```
proctype factorial (chan in_factorial int n )
{
    chan ret_factorial = [0] of {int};
    if
    :: (n == 0) ->    in_factorial ! 1;
    :: else ->    int result ;
    run factorial( ret_factorial, n - 1 );
    ret_factorial ? result
    in_factorial ! n * result;
    fi ;
}
proctype main ()
{
    chan ret_factorial = [0] of {int};
    int num ;
    int f ;
    run factorial( ret_factorial, num );
    ret_factorial ? f
}
```

Exemple2 : Fonction traitant un tableau à plusieurs dimensions

Code C

```
void Transpose()
{
  int tab1 [2][5];
  int tab2 [5][2];

  tab1[1][1]=2;

  int i,j;
  int c =0;
  for (i=0;i<2;i++)
    for (j=0;j<5;j++)
      {
        tab1[i][j]= c;
        c++;
      }

  for (j=0;j<2;j++)
    for (i=0;i<5;i++)
      tab2[i][j]=tab1[j][i];

}
```

Promela

```
proctype Transpose ()
{
  typedef line_tab1{int col[5]};
  line_tab1 tab1[2];
  typedef line_tab2{int col[5]};
  line_tab2 tab2[2];

  int i , j ;
  int c = 0;
  i=0 ;
  do
    ::!(i < 2)-> Break;
    :: else -> j=0 ;
      do
        ::!(j < 5)-> Break;
        :: else -> tab1[i].col[j] =c;
          c++;
          j++ ;
        od;
        i++ ;
      od;
  od;
```

```
j=0 ;
do
  ::!(j < 2)-> Break;
  :: else -> i=0 ;
      do
        ::!(i < 5) -> Break;
        :: else -> tab2[i].col[j] =tab1[j].col[i] ;
                    i++ ;
      od;
  j++ ;
od;

}
init {
  chan ret_ Transpose = [0] of { bit };
  run main(ret_ Transpose);
  ret_ Transpose? 0
}
```

Exemple 3 : Fonction de Fibionacci

Code C

```
int Fibionacci (int n)
{
  if (n <= 2) return 1;
  else
  {
    int a, b;
    a = Fibionacci (n - 1);
    b = Fibionacci (n - 2);
    return a + b;
  }
}

int main()
{
  int result;
  result = Fibionacci (5);
  return 0;
}
```

Promela

```
proctype Fibionacci (chan in_Fibionacci:int n )
{
    chan ret_Fibionacci = [0] of {int};
    if
        :: (n <= 2)->    in_Fibionacci ! 1;
        :: else ->    int a , b ;
    run Fibionacci( ret_Fibionacci, n - 1 );
    ret_Fibionacci ? a
    run Fibionacci( ret_Fibionacci, n - 2 );
    ret_Fibionacci ? b
    in_Fibionacci ! a + b;
    fi ;
}

proctype main (chan in_main)
{
    chan ret_Fibionacci = [0] of {int};
    int result ;
    run Fibionacci( ret_Fibionacci, 5 );
    ret_Fibionacci ? result
    in_main ! 0;
}

init {
    chan ret_main = [0] of { bit };
    run main(ret_main);
    ret_main ? 0
}
```

Conclusion

La phase d'implémentation du CProl réalisée, nous avons engagé un nombre de jeux de tests soigneusement établis, qui est en vérité un ensemble de codes résumant tous les aspects du sous-ensemble C choisi. La translation du code Promela générée pour chaque exemple était correcte. Ce qui a permis de lancer la vérification avec le Spin model-checker.