**1 . *Le langage.*** Athapascan-1 est un langage parallèle (librairie C++) permettant la construction distribuée et à la volée du flot de données associé à l'exécution d'un programme. Le grain de parallélisme est explicite :

- le grain de calcul est la **procédure** (`task`) ;
- les procédures peuvent accéder à des **objets partagés** (`shared<T>` où `T` représente le grain de donnée).

Les appels de procédure (`fork`) seront synchronisés par les accès qui sont réalisés sur les objets partagés :

- **lecture** uniquement (`R`),
- **écriture** uniquement (`W`),
- **mise à jour** (`R_W`)
- **accumulation** (`CWf`) ; écriture concurente, par accumulation des différentes valeurs écrites sur la valeur initiale. L'accumulation est définie au sens d'une fonction `f` supposée par le système associative et commutative.

La sémantique des accès en mémoire partagée est définie par un ordre lexicographique sur le programme (';') (intérieur d'abord, profondeur après). Le langage est fortement typé et sans effet de bord ; les objets partagés peuvent être passés par référence (*accès différés*). Lors de l'appel d'une tâche, des restrictions sur les liaisons entre paramètre formel - paramètre effectif garantissent que :

- La consistance est *séquentielle* (pour les accès précédents),
- Il existe un ordonnancement *séquentiel non-préemptif* des tâches,
- Dès que l'ensemble des paramètres requis par une tâche sont prêts, une exécution séquentielle non-préemptive de cette tâche respecte la sémantique.

Ainsi, des exécutions théoriquement efficaces en temps et mémoire sont possibles pour de nombreux programmes.

**2. *Exemple.*** Deux versions d'un programme récursif simple. (a) est le programme de la démonstration. Dans (b), l'écriture par accumulation permet de minimiser les synchronisations et réduire le nombre de tâches.
Le mode d'accès d'un objet partagé (`r`, `w`, ...) est décrit dans le proptotype de la tâche.

```
task fibo( int n, shared( w )<int> res ) {
 if( n<2 )
   res.write( n ) ;
 else {
   shared<int> x, y ;
   fork fibo( n-1, x ) ;
   fork fibo( n-2, y ) ;
   fork sum( x, y, res ) ;
 }
}

task sum( shared( r )<int> x,
          shared( r )<int> y,
          shared( w )<int> z ) {
  z.write( x.read() + y.read() ) ;
}

void main (int argc, char *argv) {
  shared<int> x ;
  fork fibo ( atoi(argv[1]), x) ;
}
```

(a)

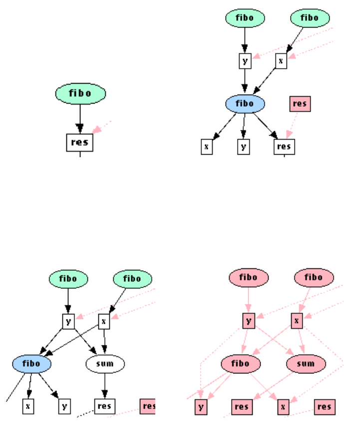
```
struct add {
  void operator()( int& x, const int& y ){
  x += y ;
 }
};

task fibo( int n,
           shared( cwf<add> )<int> res ) {

  if( n<2 )
    res.cumul( n ) ;
  else {
    fork fibo( n-1, x ) ;
    fork fibo( n-2, y ) ;
  }
}

void main (int argc, char *argv) {
  shared<int> x (0) ;
  fork fibo ( atoi(argv[1]), x) ;
}
```

(b)



(c)

**3. *Déroulement de l'exécution.*** À la volée, les tâches restant à exécuter et leurs dépendances sur les versions des objets partagés définissent l'état du système. Cet état est distribué et représenté par un graphe. Les figures (c) donnent quelques étapes de l'évolution de ce graphe pour le programme (a).

**4. *Ordonnancement et performances.*** Le graphe permet de garantir les synchronisations lecture-écriture lors de l'exécution, par détection distribuée de fin d'accès. Un module séparé décide du site et du démarrage d'une tâche : une bibliothèque extensible comportant différents ordonnancements y est associée.
Le tableau suivant donne les performances du programme (b) pour `fibo(40)` sur différentes machines. La récursivité a été stoppée à 25 (environ 1ms) ; le nombre de tâches est de 35422. Ces tâches sont exécutées par un nombre fixé de threads sur chaque machine et ordonnancées localement par Athapascan-0. Les temps sont exprimés en seconde.

| | IBM-SP 1× 1 proc/nœud | NOW 1× 1 proc/nœud | SMP 1× 4 proc/nœud | IBM-SP 4 × 1 proc/nœud | NOW 4× 1 proc/nœud |
|---|---|---|---|---|---|
| Séquentiel | 43.30 | 39.90 | 22.60 | 43.30 | 39.90 |
| Ath-séquentiel | 43.84 | 39.94 | 23.29 | 43.84 | 39.94 |
| 1 Thread | 106.45 | 101.79 | 24.45 | 27.30 | 27.76 |
| 2 Threads | 80.63 | 71.25 | 16.29 | 20.99 | 21.51 |
| 4 Threads | 67.16 | 59.93 | 13.94 | 18.34 | 18.40 |
| 8 Threads | 61.66 | 54.16 | 12.98 | 18.51 | 21.05 |

http://www-apache.imag.fr/ *Jean-Louis Roch, Gerson Cavalheiro, Mathias Doreille, François Galilée*