DEVELOPPER GUIDE

CHARLES PRUD'HOMME

This document is dedicated to developers and advanced users of CHOCO. It presents CHOCO-oriented programming rules and some common Java programming tips and tricks. We strongly encourage developers and users to read "Java Performance Tuning" [Shirazi, 2002].

Feel free complete this document.

1. Iterate

1.1. **...over domain values.** An `IntDomainVar` provides 3 main ways to iterate over its domain values.

1.1.1. *An iterator.* This is the simplest and fastest way:

```
IntDomainVar var = solver.createEnumIntVar("iv", 0, 999);
DisposableIntIterator it = var.getDomain().getIterator();
while(it.hasNext()){
   // do something...
}
it.dispose();
```

The iterator retrieves by calling `var.getDomain().getIterator()` is a `DisposableIntIterator` which means the method `next()` returns an `int` and it can be reusable if the call to `dispose()` is done. We provide this service to avoid creation of iterator any time an iteration is done. If the call to `dispose()` is correctly done, very few iterators are created during the search, and the memory can be very stable. If an iterator is not reusable (`disposable()` returns `false`), meaning it is still used, a new iterator is created and retrieved.

Exception case must also be treated, to free correctly the iterator on a `ContradictionException`:

```
DisposableIntIterator it = var.getDomain().getIterator();
try{
   while(it.hasNext()){
   // do something that can throw a ContradictionException
   }
}finally{
   it.dispose();
}
```

Keep in mind that the domain type has an impact on performance. An iteration over an enumerated domain required to check wether the compute value belong to the domain, and not only to the interval defined by the bounds.

References

[Shirazi, 2002] Shirazi, J. (2002). *Java Performance Tuning*. O'Reilly & Associates, Inc., Sebastopol, CA, USA, 2nd edition.