CnC

A highly desirable solution to the multicore software productivity problem is to introduce high-level declarative programming models that are accessible to domain experts who have deep subject matter expertise in their respective domain but are not expected to have a high level of expertise in Computer Science. In the Intel Concurrent Collections (CnC) programming model, the parallel structure of a program is described declaratively in terms of computation steps that communicate via data items that satisfy the dynamic single assignment property. The step, item, and tag collections in CnC are used to enforce two

types of constraints: data dependence, when a step produces data consumed by another step, and control dependence, when a step determines if a new step should be created.

CnC specifies the high-level coordination structure of a parallel program --- a complete program can be obtained by using a sequential or parallel imperative language to implement individual computation steps. A short introduction to the Concurrent Collections model can be found in the following article:

1. The Concurrent Collections Programming Model. Michael G. Burke, Kathleen Knobe, Ryan Newton, Vivek Sarkar. Technical Report TR 10-12, Department of Computer Science, Rice University, December 2010. To appear as a book chapter in Encyclopedia of Parallel Computing, David Padua (Ed.), Springer Verlag, 2011

The Intel CnC implementation uses C/C++ as the underlying language for CnC steps. In the Habanero Concurrent Collections project, we instead use Habanero-Java (HJ) as the underlying language for CnC steps with work in progress for step code written in Scala, Python, and Matlab. More technical details on CnC can be found in the papers in the PLDI 2009 tutorial on CnC and in the following journal article:

1. Concurrent Collections. Zoran Budimlic, Michael Burke, Vincent Cave, Kathleen Knobe, Geoff Lowney, Ryan Newton, Jens Palsberg, David Peixotto, Vivek Sarkar, Frank Schlimbach, Sagnak Tasirlar. Scientific Programming, 18:3–4, pp. 203–217, August 2010.

Finally, downloads of Habanero CnC research prototypes are available for CnC-OCR, CnC-HJ, CnC-Python, and CnC-Scala. See also DFGR and PIPES on our Publications page.