CREST
Core Research for Evolutionary Science and Technology

Japan Science and Technology Agency

Research Area : Development of System Software Technologies for Post-Peta Scale High Performance Computing

An Evolutionary Approach to Construction of a Software Development Environment for Massively-Parallel Heterogeneous Systems

Team & Group Leader:
Hiroyuki TAKIZAWA
Tohoku University

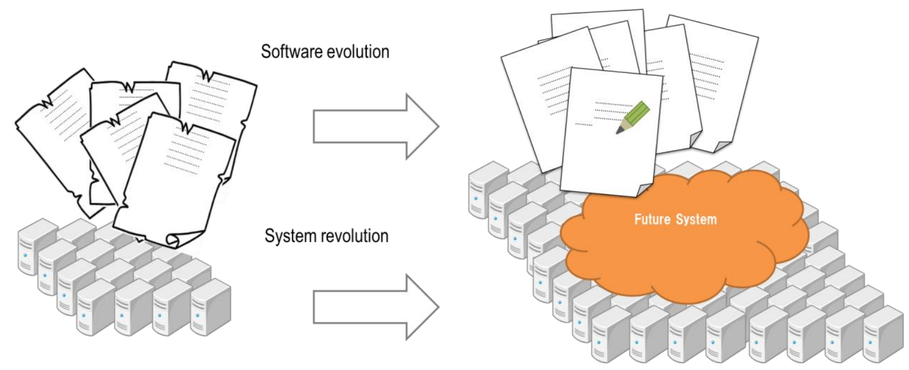
Group Leader:
Daisuke TAHAKASHI
University of Tsukuba

Group Leader:
Reiji SUDA
The University of Tokyo

Group Leader:
Ryusuke EGAWA
Tohoku University

While HPC system architectures may change revolutionarily, HPC applications can change only evolutionarily.

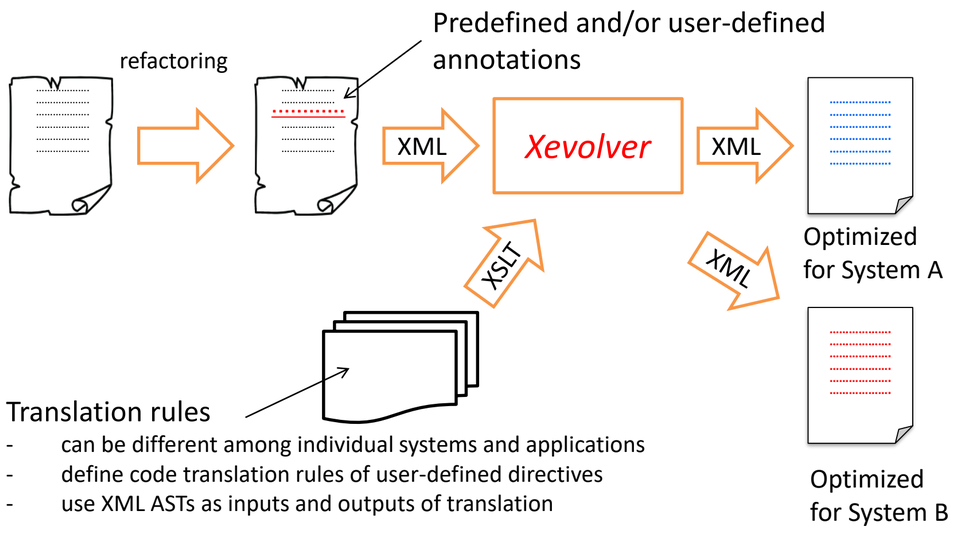
The goal of this project is to develop an extensible framework to help software evolution by achieving both high performance and high performance portability. At present, HPC applications are often optimized for their particular target systems, and hence unable to run efficiently on other systems. System-specific optimizations are essential to exploit the performance of modern HPC systems. Thus, our programming framework is designed so that it can separate those system-specific optimizations from application codes. If system-specific optimizations are written separately in external files, we can use different optimizations for different systems, resulting in high performance and high performance portability. In the programming framework, the implementation details are hidden to application programmers but visible to expert performance engineers. System-specific optimizations are abstracted in various ways such as code transformations, libraries, and domain-specific tools. As a result, system-specific optimizations are hierarchically abstracted and hence their abstractions are provided at several abstraction levels.



Separation of system-specific optimizations from HPC application codes.

For an application to fully exploit the potential of a particular system, there is no silver-bullet to completely hide everything specific to the system. Thus, there are quite a huge number of approaches for abstraction of system-specific optimizations. Yet, more or less, some code modifications are still required for various system-specific reasons. For example, abstractions such as numerical libraries are not necessarily transparent to an application code, and hence the code might need to be modified so as to use the abstractions. Such code modification is likely to be specific to a particular environment and/or a particular application. Thus, they could severely degrade the readability, maintainability and performance portability of the application. This means that we need another abstraction layer to express such special code modifications.

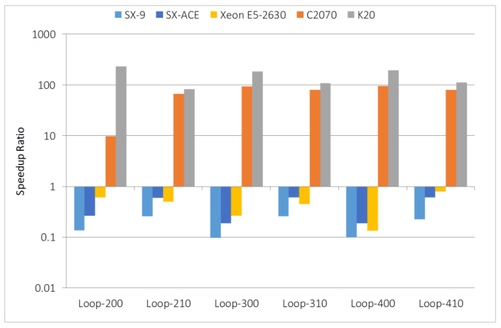
Xevolver: Code Transformation Framework, Xevtgen: Transformation Rule Generator

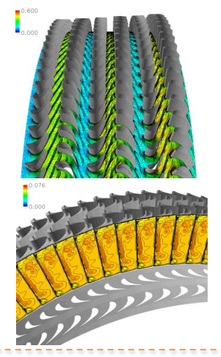


- can be different among individual systems and applications
- define code translation rules of user-defined directives
- use XML ASTs as inputs and outputs of translation

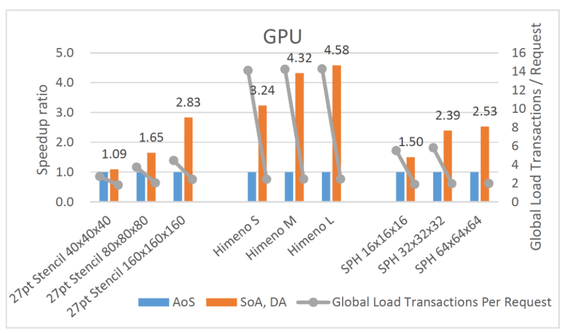
To prevent system-specific code modifications, we are developing a code transformation framework, Xevolver, which enables users to define their own code transformations for special demands of individual systems and individual applications. Instead of modifying a code by hand, Xevolver allows users to transform an application code for a particular target system, according to user-defined code transformation rules. Since the rules are written in external files, users can express system-awareness separately from an application code. By using a different code transformation for each system, one application code can be used to achieve high performance on various systems. Xevolver provides high-level interfaces and tools to define such a custom code transformation rule. Xevtgen is the most important tool to generate a code transformation rule from a simple description about the code transformation needed for system-specific code optimization. In the case of using Xevtgen, what users need to do is to simply write two versions of a code; the original version and its transformed version. Then, Xevtgen can generate a machine-usable code transformation rule that can actually translate the original code version to the transformed one. As a result, users can define their code transformation rules without any special knowledge about the techniques internally used in Xevolver. In our project, the practicality and benefit of using the Xevolver framework have been demonstrated through various case studies using real-world applications.

Loop Transformation: one of our target applications, Numerical Turbine.





Data Layout Optimization Using Code Transformations



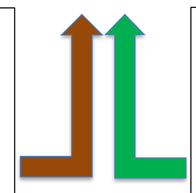
Xevtgen: Automatic Rule Generator from Code Snippets.

Tgen-variables + Before(src)

```
!$xev tgen var(i, i0, i1, p, q) exp
!$xev tgen list(body1, body2, body3) stmt

!$xev tgen src begin
!$xev loop unswitch
 do i = i0, i1
  if (p) then
   body1 = xevtgen_var
  else if (q) then
   body2 = xevtgen_var
  else
   body3 = xevtgen_var
  end if
 end do
!$xev tgen src end
```

Xev Translation Rule



Automatic Rule generation

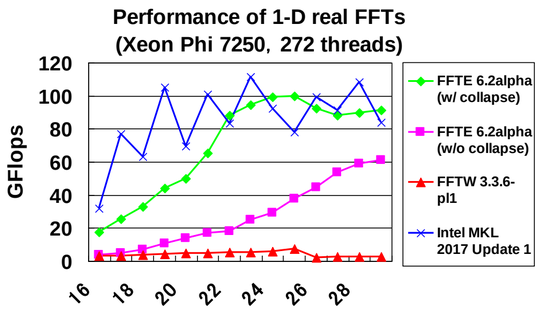
After(dst)

```
!$xev tgen dst begin
 if (p) then
  do i = i0, i1
   body1 = xevtgen_var
  end do
 else
!$xev concat_if
!$xev loop unswitch
 do i = i0, i1
  if (q) then
   body2 = xevtgen_var
  else
   body3 = xevtgen_var
  end if
 end do
end if
!$xev tgen dst end
....
```

FFTE: Numerical Libraries and Algorithms

Important numerical libraries and their algorithms need to be redesigned for massively parallel heterogeneous systems.

- Fast Fourier Transform (FFT)
- Algebraic Multigrid Method (AMG)
- Mixed-precision BLAS
- GNU Multiple Precision Arithmetic Library(GMP)



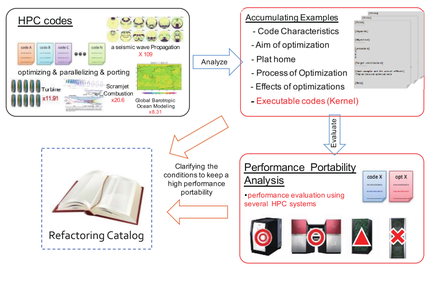
- FFTE is a Fortran subroutine library for computing the **fast Fourier transform (FFT)** in one or more dimensions.
- Includes real, complex, mixed-radix and parallel FFTs.
- Supports OpenMP, MPI, OpenMP + MPI, CUDA Fortran + MPI
- Supports a 2-D decomposition for parallel 3-D FFTs.
- Programs are vectorized by using SIMD instructions (Intel AVX2/AVX-512) and parallelized by using OpenMP.

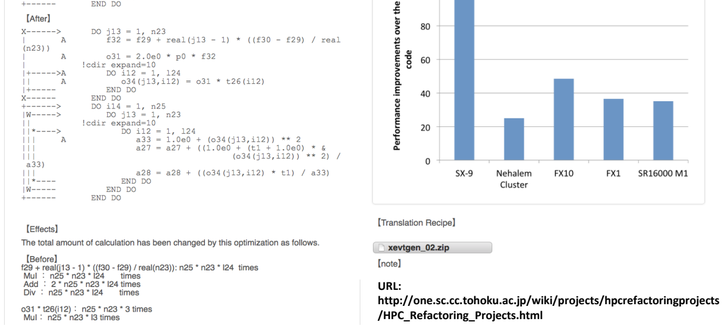
```
      complex*16 x(n1,n2),y(n2,n1)     !$omp do
!$omp parallel                              do i=1,n1
!$omp do collapse(2) private(i,j,jj)        call in_cache_fft(y(1,i),n2)
      do ii=1,n1,nb                         end do
        do jj=1,n2,nb                       ...
          do i=ii,min(ii+nb-1,n1)       !$omp end parallel
            do j=jj,min(jj+nb-1,n2)
              y(j,i)=x(i,j)
            end do
          end do
        end do
      end do
```

To expand the outermost loop, the double-nested loops are collapsed into a single-nested loop.

HPC Refactoring Catalog

Aiming at supporting smooth code migration and optimization among HPC systems, the database of system-aware code optimization patterns named HPC refactoring catalog has been developed. Currently, the catalog contains over 40 system-aware code optimization patterns and mainly used as a guideline for code migrations and optimizations. Besides, the catalog also provides code transformation rules for Xevolver.



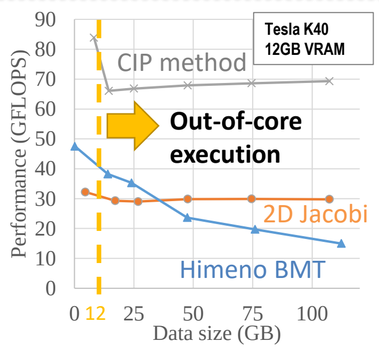


URL: http://one.sc.cc.tohoku.ac.jp/wiki/projects/hpcrefactoringprojects/HPC_Refactoring_Projects.html

PACC: Pipelined Accelerator

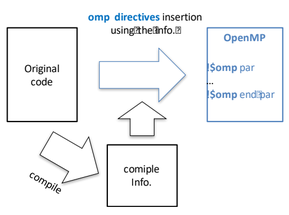
PACC is an extension of OpenACC directives for **out-of-core** stencil computation with **temporal blocking.**

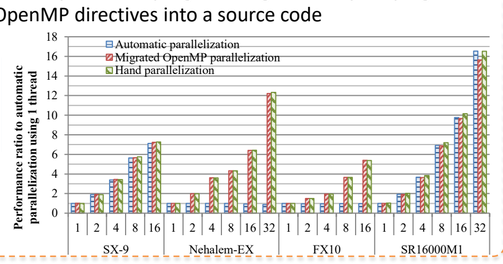
```
#pragma pacc init
#pragma pacc pipeline targetinout(p,q) ¥
   size([0:X][0:Y]) halo([1:1][1:1]) async
for(t=0;t<T; t++){
  #pragma pacc loop dim(2)
  for(x=1;x<X-1;x++)
    #pragma pacc loop dim(1)
    for(y=1;y<Y-1;y++)
      q[x][y] = p[x-1][y] + ...  ;
...
}
```



autoOMP

- a tool for automatic insertion of OpenMP directives using **compiler messages of automatic optimizations**.
- the compiler messages of a particular compiler are helpful even for other systems
- realizes high performance portability with low programming efforts by keeping these compiler messages as OpenMP directives into a source code





Other Software

Xev-GMP
- Directive-based automatic code generation for multiple-precision C code

XevWeb
- Web interfaces for Xevolver, Xevtgen, Xev-GMP.

SpMV Library
- SpMV library for GPUs

FXTPACK
- A set of routines for Fast Orthogonal Function Transforms

ASL Wrapper
- An FFT wrapper library to use ASL FFT library

AMGS
- A liberary for Algebraic Multigrid Methods

Software available http://xev.sc.cc.tohoku.ac.jp/software/ E-mail: xevolver@xev.sc.cc.tohoku.ac.jp