Structured parallel programming: a view from Pisa
M. Danelutto
Dept. Computer Science – Univ. of Pisa

Algorithms skeletons
- Concerns: new, portable, reusable
- Parallel, scalable, composable
- Cuts language constructs
- Multithreading, tasking, load balancing
- Concurrent, composable stages
- Programmable communication

Template based implementation model
- Skeletons + intra-template, target, graph, perf model
- Skeleton Library
- Skeleton Assignment: looking for the "best" template to com-posite with a skeleton in the skeleton tree
- Resource (or "soft"), matched resource assignment method according to the performance models of the skeletons templates

Power management
- Performance & Energy are major concern
- MDF: get energy & voltage
- Specific cases: battery, noise
- HPF: other criteria
- Applications: energy & voltage

Projects (Pisa)

Mechanisms and policies modeled
- Intra-monitor application operations (performance)
- Inter-monitor propagation of policies (optimization)
- To implement policies (pattern mechanisms)

Fastflow

Power management
- Template tree = meta data flow graph
- Input task = input skeleton to fetch instance of the MDF graph
- Perform transformation ← transform templates in the template pool
- Optimize node, add parallel

MDF implementation model
- Skeleton tree = meta data flow graph
- Input task = input skeleton to fetch instance of the MDF graph
- Perform transformation ← transform templates in the template pool
- Optimize node, add parallel

History (Pisa)
Developed within COREGRID & GREECEP (FP6)
- Automatic manager guided by each skeleton
- MAPS (Multiple active strategies, RISC style, machine and plan)
- Actuation (reactive)
- Single concern management
- Multiple concern management (centralized manager or decentralized coordination)
- Free provided (IA fundamentally & dynamically evolved)

• Skeletons: single principle techniques
- Тemplates: single principle techniques
- Programs: Pisa contributed patterns

System programmer Application programmer
Skeleton instance (with perf models)
Skeleton library
Target architecture
Optimizations

Overall view

O/I types, worker code

And more ...
- Modelling performance and energy
- Model-driven based approach (with Prof. Antonio Botta and Danieli Visaggio)
- High-level, high-level high-level
- Field computing patterns modeling

Paralleled design patterns
- Matrix, vector, Montage: parallel programming book 2004
- Recursive: parallelization
- Strided data access: recurrence forwarding, algorithmic space, implementa- tion, structure space, machine
- Parallel different models of the art frameworks: Intel TBB, Microsoft TPL
- Machine-to-machine and software optimization
- Key concepts: core parallel programming

Fastflow (Pisa)
- Dynamic & efficient, lock free cores (G raise)
- Memory, distributed, load balancing, steering, parallel, parallel patterns
- Dynamic middleware (CPS, etc.), distributed clients (experimental)
- Software selection, offloading selection (remote core, to device store)
- Performance analysis & debug of specified parallel patterns

RPL
- Parallel loop domains: loops and reduce a single pattern ex- pression, recursively, up to sublocally
- Parallel loop domains: loops and reduce a single pattern expres- sion, recursively, up to sublocally
- Recursive loop domains: apply all rewriting, several times, to obtain a number of parallel patterns
- Recursively parallel loop domains: apply all rewriting, several times, to obtain a number of parallel patterns
- Recursive loop domains: apply all rewriting, several times, to obtain a number of parallel patterns
- Recursive loop domains: apply all rewriting, several times, to obtain a number of parallel patterns

Sample FF code

Credits

Sample FF code

https://www.sharelatex.com/project/57305065337a19b8597