

Parallel Programming and Heterogeneous Computing

E1 - Outlook: Problem Classes

Max Plauth, *Sven Köhler*, Felix Eberhardt, Lukas Wenzel, and Andreas Polze
Operating Systems and Middleware Group

The Landscape of Parallel Computing Research: A View from Berkeley



*Krste Asanovic
Ras Bodik
Bryan Christopher Catanzaro
Joseph James Gebis
Parry Husbands
Kurt Keutzer
David A. Patterson
William Lester Plishker
John Shalf
Samuel Webb Williams
Katherine A. Yelick*

Berkeley Dwarfs

**ParProg20 E1
Problem Classes**

Sven Köhler

Chart 2

A View From Berkeley

- Sources
 - EEMBC benchmarks (embedded systems), SPEC benchmarks
 - Database and text mining technology
 - Algorithms in computer design and graphics, machine learning
 - Original “7 Dwarves” for supercomputing [Colella]
- “Anti-benchmarks”
 - Dwarfs are not tied to code or language artifacts
 - Can serve as understandable vocabulary across disciplines
 - Allow feasibility study of hardware and software design
 - No need to wait for applications being developed
- Relevance of single dwarfs widely differs
- One dwarf may be implemented based on an other one
- Reference implementations for different platforms exist

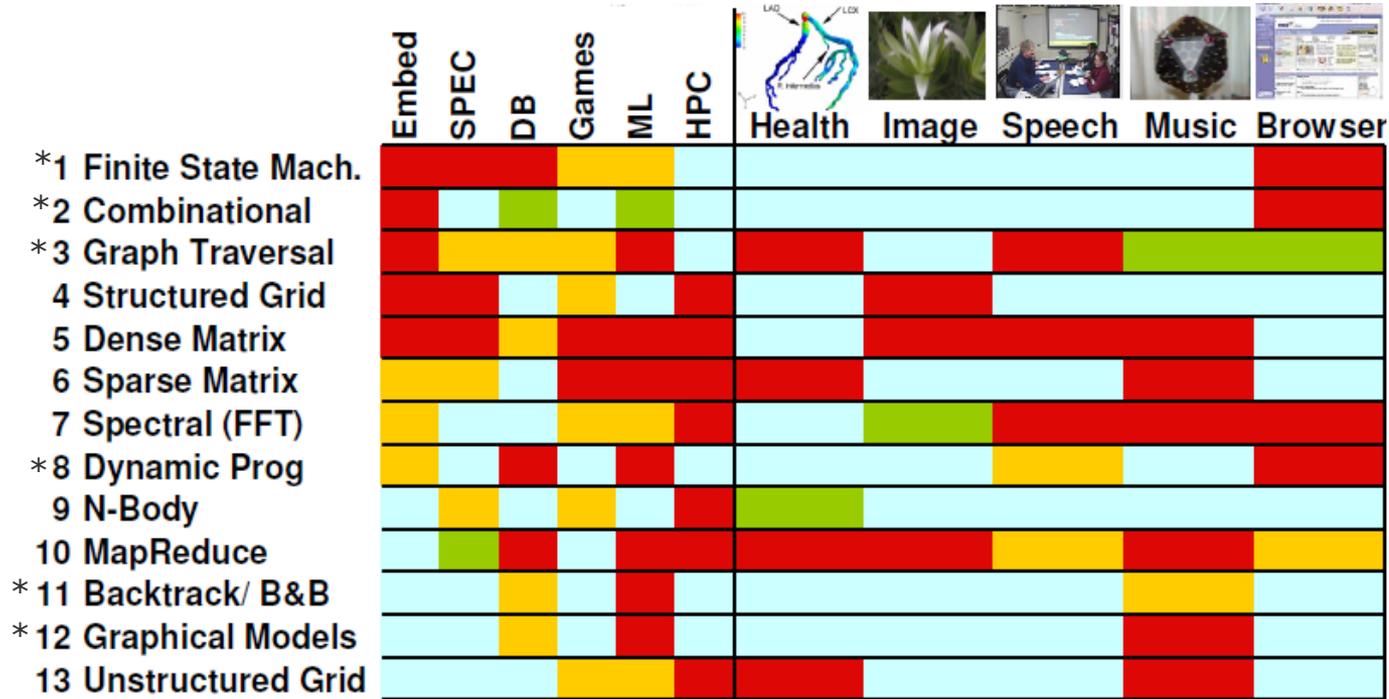
**ParProg20 E1
Problem Classes**

Sven Köhler

Chart **3**

Dwarf Popularity

= How Compelling Apps Relate To Dwarfs



Hot → Cold

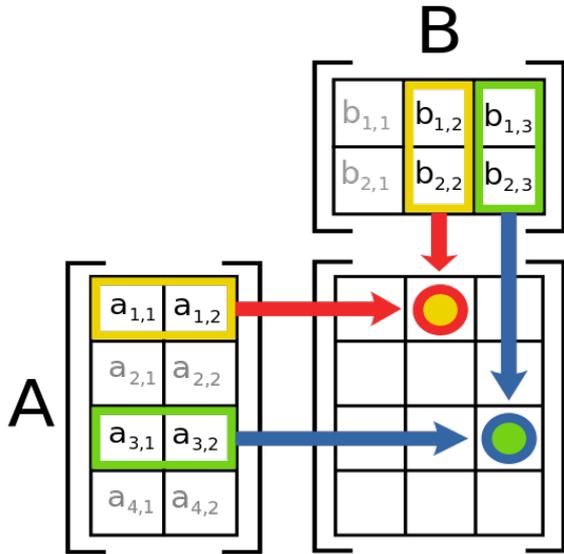
ParProg20 E1
Problem Classes

Sven Köhler

Chart 4

* added later

Dwarf 1: Dense Linear Algebra



Classic vector and matrix operations

```
do i=1,n
  do j=1,n
    do k=1,n
      a(i,j) = a(i,j) + b(i,k)*c(k,j)
```

Frequent operation in computer graphics
and as training step in machine learning

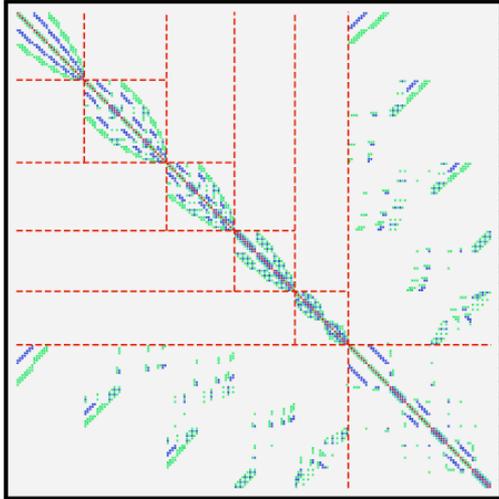
$$C = A \times B$$

**ParProg20 E1
Problem Classes**

Sven Köhler

Chart 5

Dwarf 2: Sparse Linear Algebra



Operations on a sparse matrix (lots of zeros)

```
do i=1,n
  do j=row_start(i),row_start(i+1)-1
    y(i) = y(i) + val(j)*x(col_index(j))
```

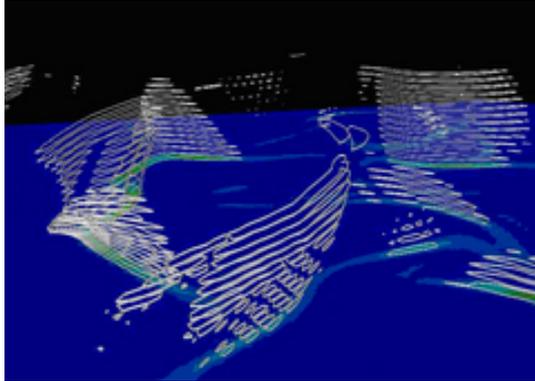
Complex data-dependency structure
Common in e.g. in graph problems.

ParProg20 E1
Problem Classes

Sven Köhler

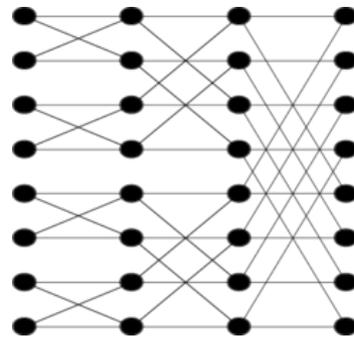
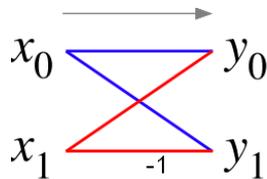
Chart 6

Dwarf 3: Spectral Methods

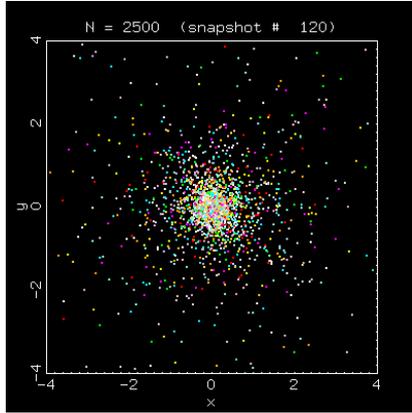


Data is converted into other domains, which means multiple stages with inter-dependent data access patterns.

Common ML data preparation step, or used in signal processing.

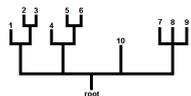
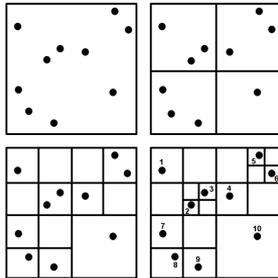


Dwarf 4: N-Body Methods



Calculations on interactions between
Many discrete points.

Large number of independent calculations
in a time step, followed by wide communication.

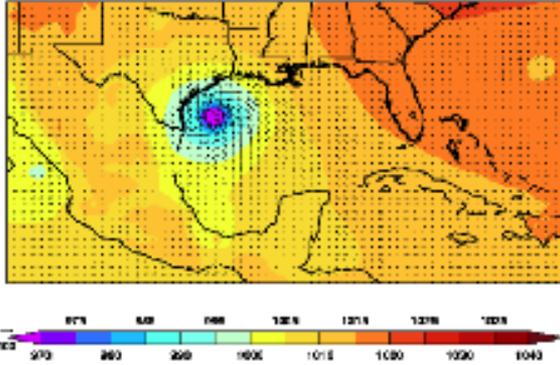


**ParProg20 E1
Problem Classes**

Sven Köhler

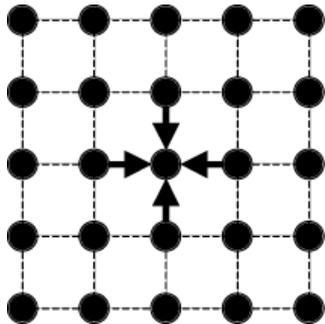
Chart 8

Dwarf 5: Structured Grid



Data as a regular multidimensional grid:
access is regular and statically
determinable (strided).

Computation is sequence of grid updates
(all points are updated using values from a
small neighborhood).



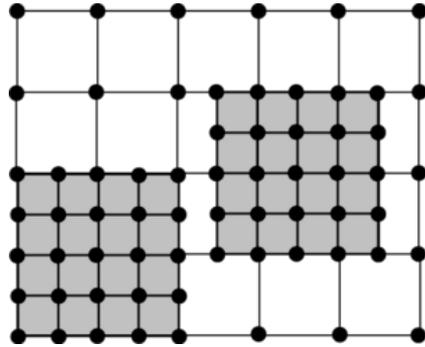
Typical Application: Weather simulations

**ParProg20 E1
Problem Classes**

Sven Köhler

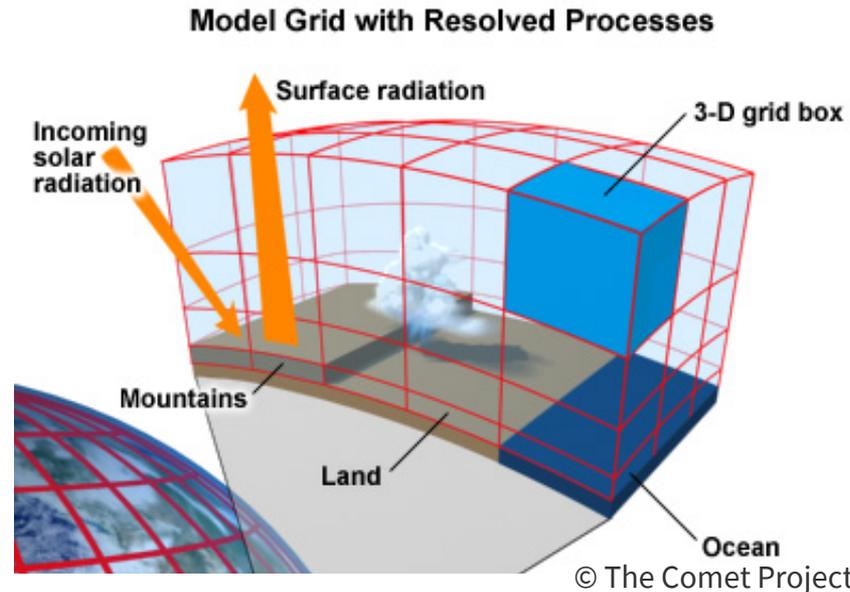
Chart 9

Dwarf 5 Variant: Adaptive Mesh Refinement



Overlaying higher-resolution grids across areas of interest. Requires complex indexing and difficult communication across nodes.

Example:
Modular Ocean Model

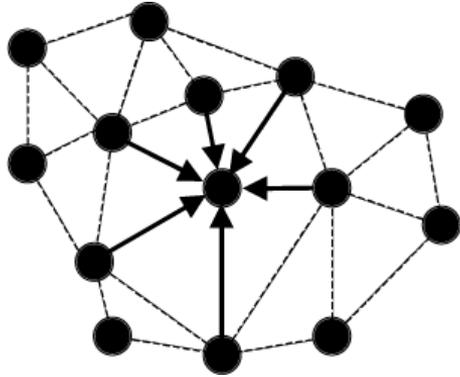


**ParProg20 E1
Problem Classes**

Sven Köhler

Chart 10

Dwarf 6: Unstructured Grid



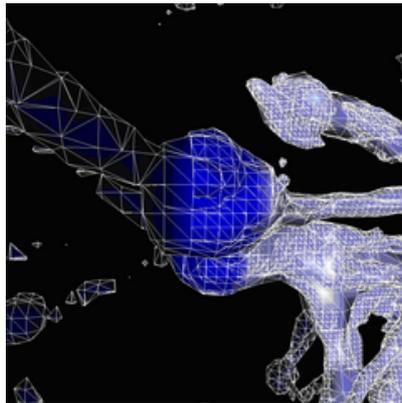
Elements update neighbors in irregular mesh/grid with static or dynamic structure

Problematic data distribution and access requirements, usually indirection by tables.

$$A'[B[C[i]]] = f(A[B[C[i+1]]] + A[B[C[i+2]]] + A[B[C[i+3]]])$$

Modelling domain (e.g. physics engine)

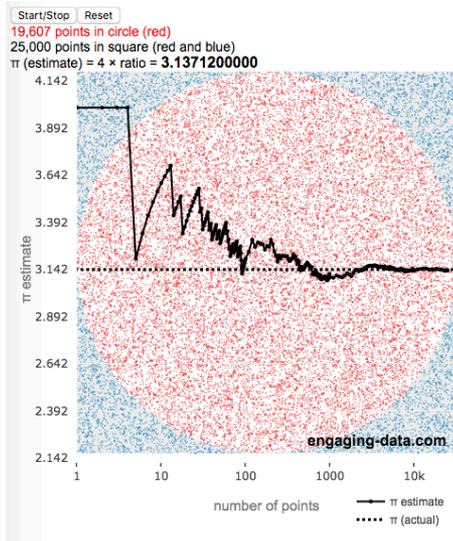
- Mesh represents surface or volume
- Entities are points, edges, faces, volumes, ...
- Applying tension, temperature, pressure



**ParProg20 E1
Problem Classes**

Sven Köhler

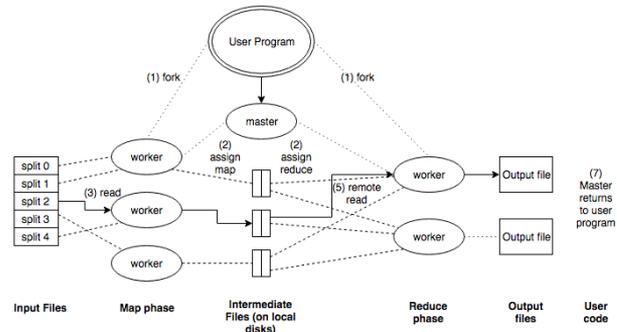
Dwarf 7: MapReduce (= "Monte Carlo")



Repeated independent execution of a function (e.g. RNG, map function), results aggregated.

Examples:

Monte Carlo Pi, BOINC (SETI@home), Optimization Protein Structure Prediction



**ParProg20 E1
Problem Classes**

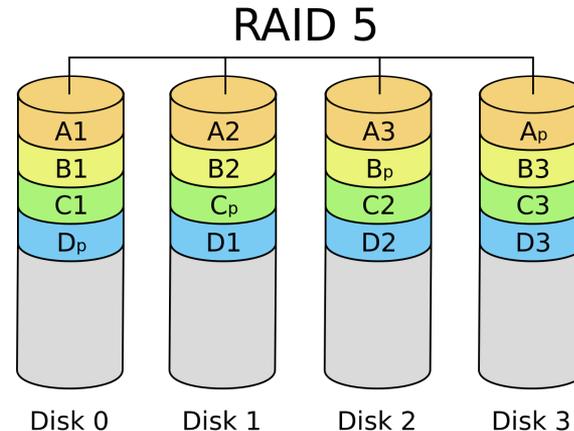
Sven Köhler

Chart 12

Dwarf 8: Combinational Logic*

- AND, OR, XOR, ...
- Exploit bit-level parallelism for high throughput
- Simple operations on very large amounts of variable-word-length data
- Parallel Mapping algorithms may be broken into data pipelines:
 - each processor executes part of the pipeline and then passes the data to the next processor
- Special-purpose hardware (or FPGAs)

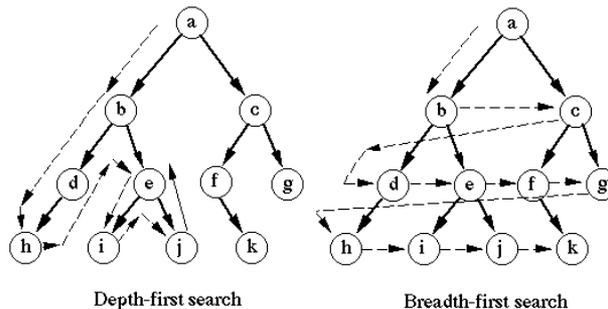
- Examples:
 - Networks and file systems: checksums, RAID
 - Data mining: population count, finding the number of 1s in a word



Dwarf 9: Graph Traversal*

- Traverse a number of objects and examine their characteristics **once**
- Usually indirect lookups and little computations
- Variation: searching
- Pointer chasing without much chance for more efficient processing

- Possible Optimizations (seldom feasible):
 - There may be locality in accesses to the graph (update graph storage)
 - There may be some processing per node that can reduce the effective cost of finding later nodes



Dwarf 10(*): Dynamic Programming

Dynamic programming matrix:

		j → (sequence y)								
		0	1	2	3	4	5	6	7	8 = N
			T	G	C	T	C	G	T	A
i ↓ (sequence x)	0	0	-6	-12	-18	-24	-30	-36	-42	-48
	1 T	-6	5	-1	-7	-13	-19	-25	-31	-37
	2 T	-12	-1	3	-3	-2	-8	-14	-20	-26
	3 C	-18	-7	-3	8	2	3	-3	-9	-15
	4 A	-24	-13	-9	2	6	0	1	-5	-4
	5 T	-30	-19	-15	-4	7	4	-2	6	0
	M = 6 A	-36	-25	-21	-10	1	5	2	0	11

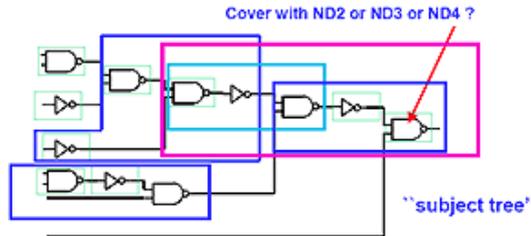
Compute optimal solutions by combining optimal, yet simpler overlapping subproblem solutions (typically use a table to avoid recomputation)

Examples:

circuit design, DNA sequence matching (Needleman–Wunsch), Viterbi, Knapsack, ...

Optimum alignment scores 11:

T	-	-	T	C	A	T	A
T	G	C	T	C	G	T	A
+5	-6	-6	+5	+5	-2	+5	+5

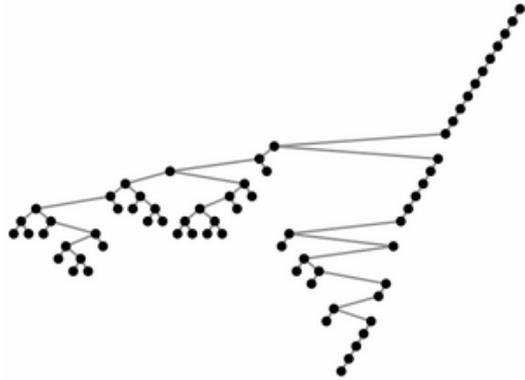


**ParProg20 E1
Problem Classes**

Sven Köhler

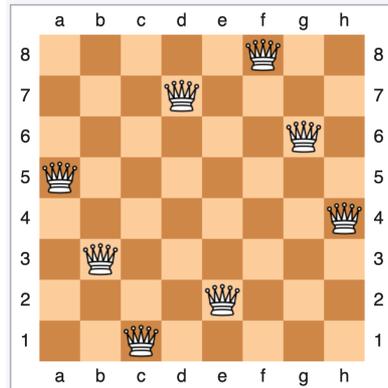
Chart 15

Dwarf 11(*): Branch-and-Bound



Global optimization problem in very large search space:

- Branches into subdivisions
- Rules out infeasible regions to optimize execution time and energy consumption



Examples:

Integer Linear Programming, Boolean Satisfiability, Combinatorial Optimization, Traveling Salesman, Constraint Programming, N-Queens

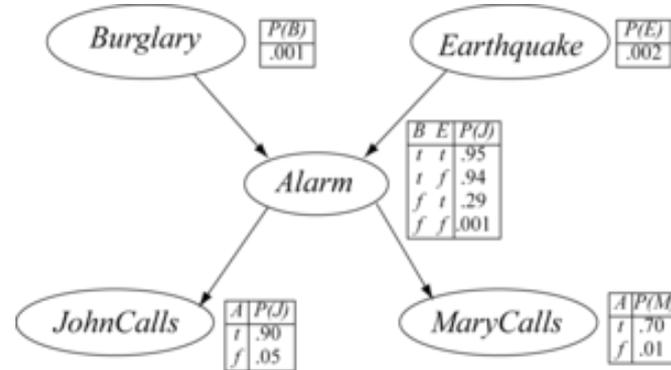
**ParProg20 E1
Problem Classes**

Sven Köhler

Chart **16**

Dwarf 12(*): Graphical Models

- A graph in which nodes represent variables, and edges represent conditional probabilities
- Bayesian networks, Hidden Markov models, neural networks
- Examples: AI, machine learning speech and image recognition
- Evaluation is a form of *Graph Traversal*, or *Dense-Linear Algebra*
- **Uniprocessor Mapping:**
 - Probabilistic aspect -> small amount of computation per node
 - Processing many observations and updating variables accordingly
- **Parallel Mapping:**
 - May be evaluated multiple times for a single problem -> Update conflicts possible
 - Simple: many graphical models can be evaluated for a single input

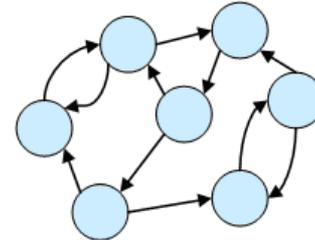


Dwarf 13(*): Finite State Machines

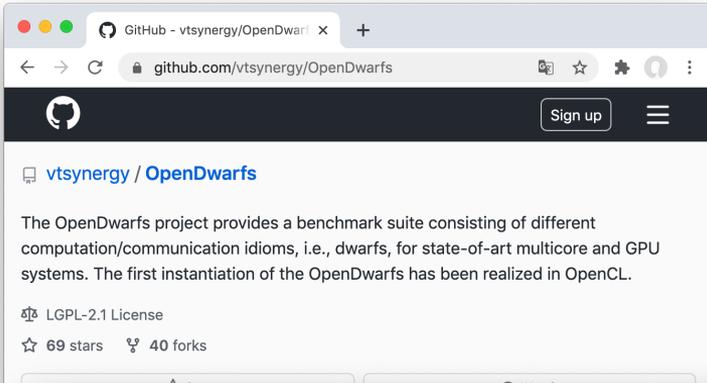
- Interconnected set of states, initial state, input, transitions (based on current inputs and state), output (based on current inputs and state)

- Parallelism in the computation of state transitions
- Decomposing into multiple state machines possible
 - Smaller and simpler
 - Combined states and outputs functionally mimic the original
 - Communication/synchronization required

- Issue: Wasted resources mapping 1 state = 1 thread (just one state possible), may not justify communication overhead
- Optimization: Decomposition, multiple FSM at the same time (case-statements within)



Exemplary Reference Implementations



GitHub - vtsynergy/OpenDwarfs

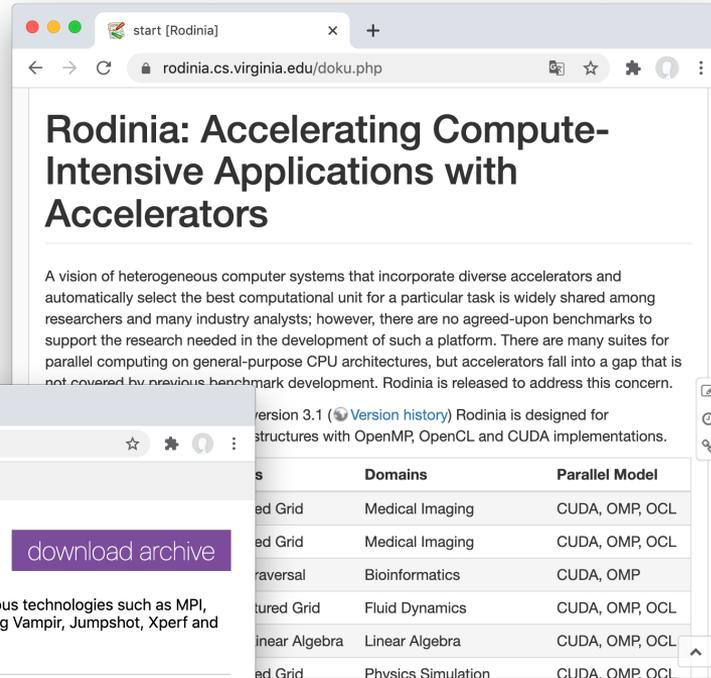
Sign up

vtsynergy / OpenDwarfs

The OpenDwarfs project provides a benchmark suite consisting of different computation/communication idioms, i.e., dwarfs, for state-of-art multicore and GPU systems. The first instantiation of the OpenDwarfs has been realized in OpenCL.

LGPL-2.1 License

69 stars 40 forks



start [Rodinia]

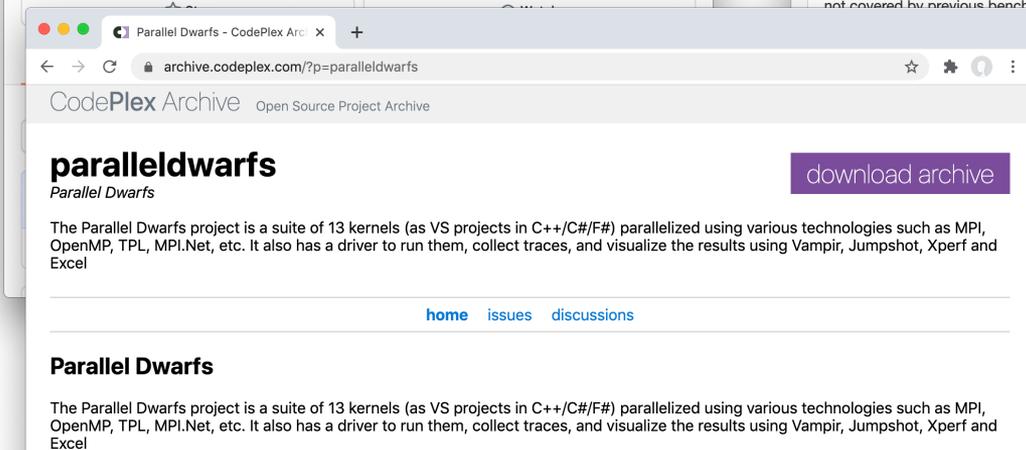
rodinia.cs.virginia.edu/doku.php

Rodinia: Accelerating Compute-Intensive Applications with Accelerators

A vision of heterogeneous computer systems that incorporate diverse accelerators and automatically select the best computational unit for a particular task is widely shared among researchers and many industry analysts; however, there are no agreed-upon benchmarks to support the research needed in the development of such a platform. There are many suites for parallel computing on general-purpose CPU architectures, but accelerators fall into a gap that is not covered by previous benchmark development. Rodinia is released to address this concern.

version 3.1 (Version history) Rodinia is designed for structures with OpenMP, OpenCL and CUDA implementations.

	Domains	Parallel Model
ed Grid	Medical Imaging	CUDA, OMP, OCL
ed Grid	Medical Imaging	CUDA, OMP, OCL
raversal	Bioinformatics	CUDA, OMP
ured Grid	Fluid Dynamics	CUDA, OMP, OCL
inear Algebra	Linear Algebra	CUDA, OMP, OCL
ed Grid	Physics Simulation	CUDA, OMP, OCL



Parallel Dwarfs - CodePlex Archive

archive.codeplex.com/?p=paralleldwarfs

CodePlex Archive Open Source Project Archive

paralleldwarfs

Parallel Dwarfs

The Parallel Dwarfs project is a suite of 13 kernels (as VS projects in C++/C#/F#) parallelized using various technologies such as MPI, OpenMP, TPL, MPI.Net, etc. It also has a driver to run them, collect traces, and visualize the results using Vampir, Jumpshot, Xperf and Excel

[home](#) [issues](#) [discussions](#)

[download archive](#)

Parallel Dwarfs

The Parallel Dwarfs project is a suite of 13 kernels (as VS projects in C++/C#/F#) parallelized using various technologies such as MPI, OpenMP, TPL, MPI.Net, etc. It also has a driver to run them, collect traces, and visualize the results using Vampir, Jumpshot, Xperf and Excel

**ParProg20 E1
Problem Classes**

Sven Köhler

Chart 19

A D
end

**ParProg20 E1
Problem Classes**

Sven Köhler

Chart **20**