Beyond3D

C++ AMP CONTEST

Results

Winning is only half of it. Having fun is the other half.

Bum Phillips

Finding ourselves at the end of a most fascinating journey, we can but congratulate all contestants, and above all the finalists, Bernd Paradies and Veikko Eeva,

who have proven to be not only able programmers but also true gentlemen. It is our sincerest hope that irrespective of the final ranking, the main victory to be found at the end of the contest is constituted by the accrual and subsequent propagation of knowledge. The role of this document is to clarify details about the problem set that was used in the scoring, how the scoring was conducted and, as closure, introduce the final ranking.

Problem set

After some deliberation, and in order to ensure ease of verification and transparency, we have opted for using problems that are publicly available in the TSPLIB library, namely:

- a280 – a 280 node symmetrical TSP instance;
- rat575 – a 575 node symmetrical TSP instance;
- u1060 – a 1060 node symmetrical TSP instance;
- d2103 – a 2103 node symmetrical TSP instance;
- fnl4461 – a 4461 node symmetrical TSP instance;
- pla7397 – a 7397 node symmetrical TSP instance.

Contrary to the initial plan, **asymmetrical TSPs were not factored into the final grading, due to the high inconsistencies encountered in the evaluated solutions.**

Scoring

In our solver scoring, we have opted for aligning running times and differentiating on correctness. Otherwise stated, competing solutions were given a precise timeout period, and the quality of the solutions achieved during this time served as discriminant. We evaluate solution quality as %Δ versus the known optimum, where:

$$\%\Delta = \frac{solution - optimal_{solution}}{optimal_{solution}} \times 100$$

We take solution as the median value of a population of ten runs. The final score is established as a weighted average of solution qualities across the problem batch:

$$score = \frac{\sum_i w_i \times \%\Delta_i}{\sum_i w_i}, where\ w_i = \frac{size(problem_i)}{\sum_i size(problem_i)}$$

By way of consequence, the following weights were held by each problem (only first 3 figures after the decimal point shown):

a280	rat575	u1060	d2103	fnl4461	pla7397
0.017	0.036	0.066	0.132	0.280	0.465

Given that displacement from optimality is being measured, a lower overall value is preferable to a higher one. By subtracting the score (in percentage points) from the ideal of 100% (no displacement from optimum), we determine the final solver score in percentage points.

Code quality is rated based on the subjective evaluation of the contest's refereeing panel, and gauges the extent in which the coding guidelines were maintained and the manner in which C++ AMP was leveraged in the solve.

In the final score, the solver quality holds a 90% weight, whereas the code quality assessment makes up for the remaining 10%.

Measurements

We only present an abridged version of our measurements here, useful for showing the basis of the scoring. For a more comprehensive treatment, see the attached documents. Characterizations across a broader hardware spectrum can be made available on request. The code was compiled with the Visual Studio 2012 Update 1 coupled with the November Compiler CTP, with full optimization (/Ox) and AVX code generation turned on (full details about compiler flags are to be found in the detailed result files).

Running Time (seconds)					
Problem type	**Problem**	**Bernd Paradies**			
		Average	**StdDev**	**Median**	**GeoMean**
TSP	**a280**	7.3501	0.0001	7.3501	7.3501
	rat575	18.0708	0.0004	18.0709	18.0708
	u1060	52.4575	0.0020	52.4576	52.4575
	d2103	89.9360	0.0020	89.9365	89.9360
	fnl4461	89.9174	0.0016	89.9177	89.9174
	pla7397	89.8376	0.0068	89.8359	89.8376
	Problem	**Veikko Eeva**			
		Average	**StdDev**	**Median**	**Geomean**
	a280	7.3705	0.2200	7.3642	7.3675
	rat575	18.1154	0.4811	18.0648	18.1097
	u1060	52.3709	9.9132	52.4182	51.4917
	d2103	88.0621	3.9014	89.9696	87.9800
	fnl4461	89.9275	0.0335	89.9144	89.9275
	pla7397	89.7843	0.0998	89.7511	89.7843

Results

Best Tour									
Problem type	Problem			Bernd Paradies					
	Name	Known opt.	Weight	Average	StdDev	Median	GeoMean	%Δ	Weighted average
TSP	a280	2579	0.02	2701	18	2696	2701	4.52%	8.73%
	rat575	6773	0.04	7090	11	7091	7090	4.69%	
	u1060	224094	0.07	240633	1778	240624	240627	7.38%	
	d2103	80450	0.13	83499	872	83602	83495	3.92%	
	fnl4461	182566	0.28	197610	1035	197352	197608	8.10%	
	pla7397	23260728	0.47	25856400	216466	25852336	25855586	11.14%	
	Name	Known opt.	Weight	Veikko Eeva					
				Average	StdDev	Median	GeoMean	%Δ	Weighted average
	a280	2579	0.02	2661	19	2660	2661	3.12%	12.26%
	rat575	6773	0.04	7106	28	7113	7106	5.01%	
	u1060	224094	0.07	239231	1382	239189	239228	6.74%	
	d2103	80450	0.13	84151	503	84205	84150	4.67%	
	fnl4461	182566	0.28	202226	2169	202472	202216	10.90%	
	pla7397	23260728	0.47	27249076	472719	27202016	27245382	16.94%	

Based on the above breakdown, the solver quality rating for each contestant becomes:

$$\boldsymbol{Bernd\ Paradies}: \mathbf{100\%} - \mathbf{8.73\%} = \mathbf{91.27\%}$$

$$\boldsymbol{Veikko\ Eeva}: \mathbf{100\%} - \mathbf{12.26\%} = \mathbf{87.74\%}$$

In order to justify the code quality rating, we reproduce below one of the referee assessments:

"Looking at Veikko's solution, the same app written in OpenCL would be similar in length and readability. It may take some time before we start to see complicated uses of types in template libraries, which is where AMP will see its full power coming from. Bernd's solution suffers similarly and the sequences of static restrict(amp) functions are very similar to OpenCL kernels written in a .cl file. It is marginally less "single-source" than Veikko's solution, but by having similar lambda functions is slightly easier to read.

Overall I think the solutions are pretty similar and both give the appearance of being adaptations of OpenCL programs into single-source format. I'd suggest 7/10 for each for code quality."

Given that the other assessments more or less converged to this same point, each of the contestants received **7%** out of the **10%** allotted to code quality.

Final score and ranking

Given all of the above, the final scores and ranking are the following:

$$\mathbf{1^{st}}: \boldsymbol{Bernd\ Paradies} - 91.27\% \times 90\% + 7\% \times 10\% = \mathbf{82.843}\%$$

$$\mathbf{2^{nd}}: \boldsymbol{Veikko\ Eeva} - 87.74\% \times 90\% + 7\% \times 10\% = \mathbf{79.666}\%$$

The contestants are to be congratulated for their exquisite efforts. As you will soon see, once their submissions are published, they were most daring in their foray into the world of GPU accelerated TSP solving. Beyond3D would also like to thank our good friends at Microsoft and at AMD, who made the contest possible and were most supportive throughout its length.

WWW.BEYOND3D.COM

DEVELOPER.AMD.COM

BLOGS.MSDN.COM/B/NATIVECONCURRENCY