team-team

100
Most significant comments: lots of good analysis and discussion. Adequate optimizations applied. More detailed comments below.

1 (inputs) Lots of experiments with input values.

2 (cache) Fixed row/column problem. Included theoretical analysis. Results on lots of problem sizes, reporting 2x speedup. No cache optimizations attempted for memory efficient algorithms.

3 (openmp) Used parallel for directives in all 3 algorithms. Did not parallelize backtrack_implicit.

4 (parallel results) Experiments with capacity 50,000 and numitems 8,000 as well as the reverse. Large capacity speedups around 5x/10x/3x, small capacity speedups around 6x/21x/12x. Strong scaling plots are clear, good to include absolute values and speed up numbers in tables. You might consider using a log scale y-axis to make the distinctions clearer when the number of threads is high. You might also consider plotting the perfect speedup curves along with experimental data.

5 (conclusions) Lots of analysis and discussion. Theoretical analysis is good. I'm not sure I agree with the hypothesis that memory efficient algorithms scale better because of less sequential work; I think it's probably due to the smaller memory footprint and less memory bandwidth contention across the threads. You could test your hypothesis by timing the sequential parts separately. Good analysis of the limits of parallelizing the inner for loops. Good discussion of other possible optimizations.