Main Contributions:

The author found that the architectures for parallel with coarse-gained parallelism is hard for compiler to take advantage of, the paper is trying to solve this problem. To deal with this, paper introduced trace scheduling to make the path of accessing the block contain more parallelism, and discussed memory reference disambiguation to deal with the undirect references. The author also pointed out that a fast system need high bandwidth with caches. And the last part had a detailed discussion about list scheduling and operation scheduling code compiling.

My Thoughts:

First, I like the logic behind the paper. It's a pretty old paper, and doesn't really follow any format, but the structure this paper goes is from bottom to top. This kind of structure made it very clear to me, unlike some of the early paper that had a bit of everything in every part and I need to find out what's the point of each part. I like the compiler part the most, that's a detailed explanation with proper amount of example.

There are few limitations, like they didn't talk about their exact experiment result when they tell us the list scheduling compiler is faster or when they said pathing can be improved. They only got a ratio table at the last part, so we can't really know how much each step this paper made can improve the whole system. But it's still a great job.

Also, I like the detailed code example a lot. Really helped me a ton.

Positive Point:

The logic is strong and the work is great.

Negative Point:

Maybe add a table for various bandwidth can make their point stronger.

The bottom line (1=terrible, 5=fantastic): __4____ **Grade (for professor): _____**