Thread: Searching for Slots with Fewest Words

Select: All None

Message Actions Expand All Collapse All

6 Posts in this Thread 0 Unread

David Alexander Tedjopurnomo 2 days ago

Searching for Slots with Fewest Words

Hi all,

I've managed to get my program up to the point where I can solve smaller puzzles (2,3,4). However, my program froze on anything larger than that.

I've implemented the strategy in the topic "Optimization on performance". Basically, for each slot, calculate the number of words that fit into that slot. Then, pick the slot that matches with the least number of words, bind that slot to a word that it can be bound to, then recalculate the number of words for each slot again.

I tested it in the smallest puzzle (puzzle 2) and it printed out the correct results. Each slot is numbered with the # of words it can be matched to and the smallest one gets bound. Then, the next iteration all have the updated values from the recalculation.

On larger puzzles, however, I am completely stuck. Are there any suggestions to cut down the calculation time?

Best regards,
David

Reply Quote Email Author

Jamie Yung 2 days ago

RE: Searching for Slots with Fewest Words

Hey David,

I recommend a constraints-based approach. See my thread titled 'Constraints-based approach' for more info.

Reply | Quote | Email Author

Peter Schachte 2 days ago

RE: Searching for Slots with Fewest Words

The strategy you describe should be fast enough to solve all of the puzzles. What should happen is that once you've filled in a slot or two or maybe three, from there on, there should always be a slot with only one match, so there should be no further need for search. Or you should quickly find a slot with no matching words, and force backtracking to select another word to fill into the selected slot.

If that's not working for you, have you checked that your slots share variables? Ie, if two slots cross, they share a variable where they cross. If not, the whole strategy won't work because filling in a slot won't partially fill in the crossing slots.

Reply | Quote | Email Author

David Alexander Tedjopurnomo 2 days ago

RE: Searching for Slots with Fewest Words

Thank you all for the response, I solved the problem.

What I did wrong is that I generated a triple (a logical variable with three "parameters") of the number of the words that can match with a slot, the slot, and any one word to match (using setof so backtracking is possible). I think the problem with this is that when I was generating the triples for all slots, the setof will cause the backtracking to be done at this step. If a slot has 38 words, for example, the program will backtrack that amount of time, maybe even more.

I fixed this problem by getting the slot first (without using setof to get the words). Then, after I get the slot with the least amount of words, I then try to unify it with a word, retrieved using setof. Since I started with a slot that has the least amont of words, the backtracking is very quick and by the time my program reached the slot that used to have a large number of words, there are already lots of slots bound to a letter that the number of words is cut down significantly. This fix solves even the largest puzzle (puzzle 5) in seconds.

I'm not sure if this falls under constraints programming, but nonetheless, it works. I hope that this will help others facing the same conundrum.

Reply | Quote | Email Author ▲ Hide 2 replies

Diana Ruth 1 day ago

RE: Searching for Slots with Fewest Words

So how are you finding the slot with the least amount of words without retrieving the words? I'm having the same problem, so any help is much appreciated :)

Reply | Quote | Email Author

▲ Hide 1 reply

David Alexander Tedjopurnomo 1 day ago

RE: Searching for Slots with Fewest Words

I use findall, which will retrieve all the words in a list so there's no backtracking. The number of words is simply the length of this list.

Reply | Quote | Email Author

Select: All None

Message Actions | Expand All | Collapse All

← OK