Cem Birler

Simulator

Since the simulation time limit is 91 days and it takes 90 days for crops to grow, planting a crop after the 1st day is not favorable because it can't be harvested in this duration. Maximum yield would also require all of the cells to be planted. (Proof is trivial: if not every cell is planted then filling an empty cell with beans on day 1 would yield at least 10 more units of food) So simulation plants every square (x,y) with either corn or beans on day 1 and harvests everything on day 91.

Details of MCTS Implementation

This problem can be turned into MCTS problem by planting cells one at a time. An action would be planting one of the empty (not planted) cells. Based on "zero or more actions on any day" all squares can be planted on 1st day and harvested on 91st day. MCTS is used to decide which squares will be planted beans or corn. Upper Confidence Trees (UCT) is used in the MCTS implementation.

I have changed the following functions from MCTS library implemented by Paul Sinclair and experimented with Random Policy as well as with a custom SelectMoreBeans Policy (user can choose the probability of planting beans and corn as next action) Random Policy is 50-50. (https://github.com/pbsinclair42/MCTS)

getPossibleActions():
Possible actions are either to plant bean or corn on one cell at a time. So, if there are k empty cells then 2*k actions are possible from that state.

takeAction():
Fill one cell in board array with either 1 (beans) or 2 (corn)

isTerminal():
finish when there is no cell with 0 value (not planted)

getReward():
Described as in the problem statement, return sum of units of food produced at the end from all of the cells.

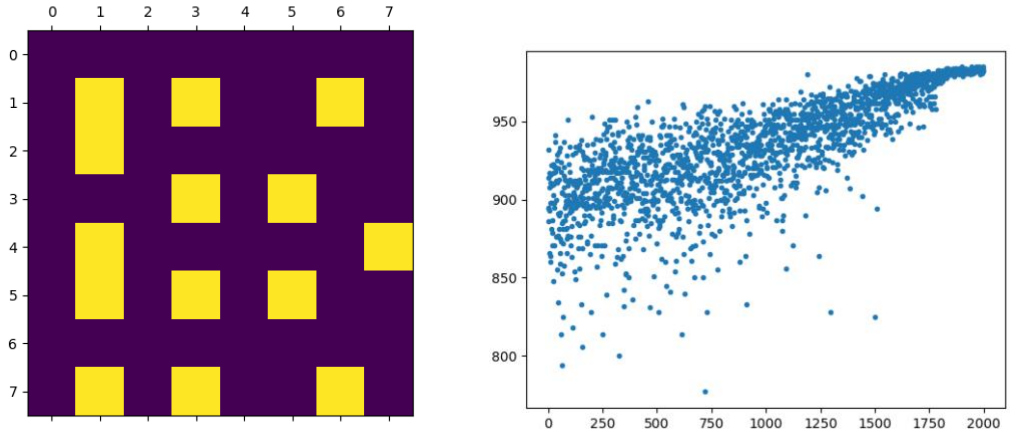
Brute force algorithm can find the optimal solution by calculating the reward for every state. Since there are 2^{n*n}states and each state take O(n*n) operation to calculate reward, total complexity is: O(n*n*2^{n*n}). For n<=5 this can be calculated disregarding the symmetries.

Exploration vs Exploitation Tradeoff

For n<10 I have chosen to use exploration constant=3.0 and run over 200000 rollouts. It is better to increase the exploration constant when both branching factor and depth of game are small, so MCTS would have a less chance of being stuck in local maximum. For n>=10 simulation is run for 1 hour with exploration constant=1.0. It is observed that for n>=11, MCTS didn't converge and Corner Square Placement Algorithm (see next page) gave the best results for n>=11 compared to MCTS. Results of

Cem Birler

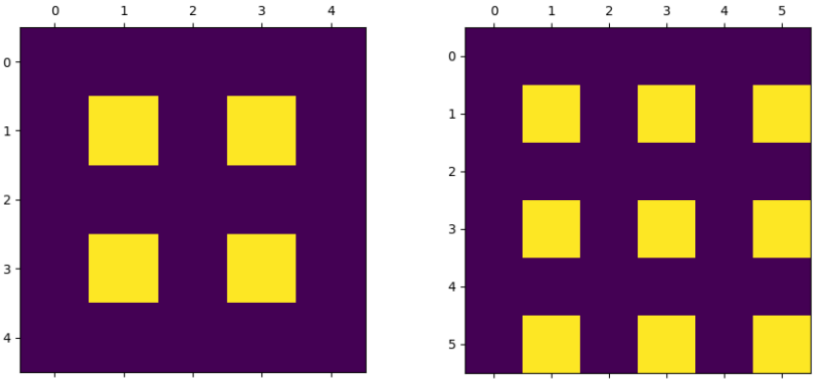
MCTS farm configuration for various n and graph of MCTS are attached in n_maxFood_method (eg: 8_986_200k) format.



(Left) For n=8, MCTS finds the configuration as the best with c=5 and in 200k rollouts. Purple squares are beans, yellow are corn. (Right) Y-axis is the units of food produced by the current rollout and X-axis is the 100th rollout. This MCTS converges to value 986.

Corner Square Placement Algorithm

Starting from the top left, on every 2x2 square area place corn on bottom right 1x1 square.



Corner Square Placement Algorithm for n=5 and n=6, yellow squares are corn and purple ones are beans.

A Different MCTS Formulation

Instead of placing beans and corn I decided that an action can be planting only corn and rest of the unplanted areas would be beans. Terminal state is reached when some pre-given number of squares are planted with corn. The advantage of this method is that the possible number of actions from a state is halved also depth is decreased, so MCTS converges faster. Inspired by the high value of Corner Square Placement Algorithm, I decided that when $n^2/4$ of the squares are planted with corn it should be the terminal state, so depth of a rollout is also ¼ of the original MCTS. This model yielded far better and faster results than previous one when run on 200000 rollouts with c=2.0 and it converged for all n<=15. This model doesn't guarantee convergence to maximum.

Cem Birler

n	Best in 200k rollouts, c=3.0	Best in 1 hour, c=1.0	CSP Inspired MCTS, c=2.0 and 200k rollouts	Corner Square Placement	Upper Bound (not tight)
6	**556**			550	594
7	761			**762**	808
8	**993**			985	1056
9	1261			**1263**	1336
10		**1551**	1543	1546	1650
11		1835	1871	**1890**	1996
12		2169	2228	**2233**	2376
13		2528	2618	**2643**	2788
14		2929	3031	**3046**	3234
15		3346	3488	**3522**	3712

Results of different MCTS formulations, Corner Square Placement algorithm and a theoretical upper bound (not tight)

Lower Bound and Upper Bound for Maximum

Claim: Maximum $16.5n^2$ units of food can be produced (Not tight upper bound).

Proof: Assume there are x beans and n^2-x corn squares. A square has at most 8 adjacent squares so, x beans can help at most 8x corn to produce extra unit of food. Since there are n^2-x corn, on average a corn square can produce (10 + 8x/(n^2-x)) units of food at most. Hence in total at most, (10 + 8x/(n^2-x))*(n^2-x) + 15x units of food can produced (assuming every bean square is also adjacent to at least one corn square), which is equal to $10n^2$ + 13x.

1) Assume that there are more corn squares than beans, then: 2x <=n^2, which gives: $10n^2$ + 13x <= $10n^2$ +$6.5n^2$ <= $16.5n^2$.
2) Assume there are more beans than corn, so in the best-case half of them are beans that yield 15 units and half of them are corns that yield 18 units, which gives at most $16.5n^2$ units food.

Note that Corner Square Placement Algorithm has $(n-1)^2/4$ corn squares that are adjacent to only beans and rest of the squares are planted beans that have at least one corn square neighbor. Hence it produces at least $15.5n^2$ units of food when n is odd and n>=7. This implies that $15.5n^2$<= Max Units of Food <= $16.5n^2$