SEARCHING ALGORITHMS

– Matee Vadrukchid –

https://ai.stanford.edu/~nilsson/OnlinePubs-Nils/PublishedPapers/astar.pdf

A Formal Basis for the Heuristic Determination of Minimum Cost Paths

PETER E. HART, MEMBER, IEEE, NILS J. NILSSON, MEMBER, IEEE, AND BERTRAM RAPHAEL

***Abstract*—Although the problem of determining the minimum cost path through a graph arises naturally in a number of interesting applications, there has been no underlying theory to guide the development of efficient search procedures. Moreover, there is no adequate conceptual framework within which the various ad hoc search strategies proposed to date can be compared. This paper describes how heuristic information from the problem domain can be incorporated into a formal mathematical theory of graph searching and demonstrates an optimality property of a class of search strategies.**

I. Introduction

A. *The Problem of Finding Paths Through Graphs*

MANY PROBLEMS of engineering and scientific importance can be related to the general problem of finding a path through a graph. Examples of such problems include routing of telephone traffic, navigation through a maze, layout of printed circuit boards, and mechanical theorem-proving and problem-solving. These problems have usually been approached in one of two ways, which we shall call the *mathematical approach* and the *heuristic approach.*

1) The mathematical approach typically deals with the properties of abstract graphs and with algorithms that prescribe an orderly examination of nodes of a graph to establish a minimum cost path. For example, Pollock and Wiebenson[1] review several algorithms which are guaranteed to find such a path for any graph. Busacker and Saaty[2] also discuss several algorithms, one of which uses the concept of dynamic programming.[3] The mathematical approach is generally more concerned with the ultimate achievement of solutions than it is with the computational feasibility of the algorithms developed.

2) The heuristic approach typically uses special knowledge about the domain of the problem being represented by a graph to improve the computational efficiency of solutions to particular graph-searching problems. For example, Gelernter's[4] program used Euclidean diagrams to direct the search for geometric proofs. Samuel[5] and others have used ad hoc characteristics of particular games to reduce

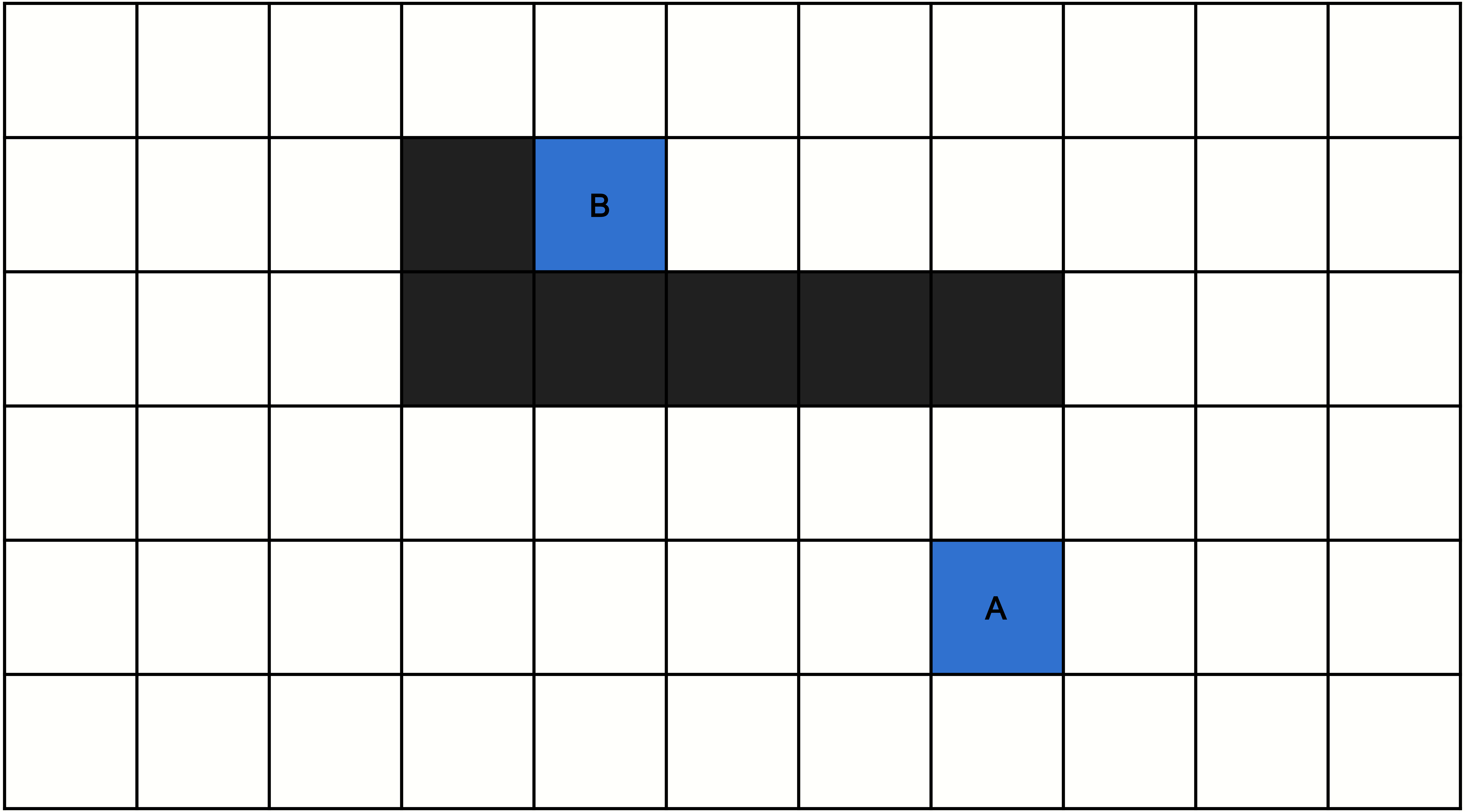
Manuscript received November 24, 1967.
The authors are with the Artificial Intelligence Group of the Applied Physics Laboratory, Stanford Research Institute, Menlo Park, Calif.

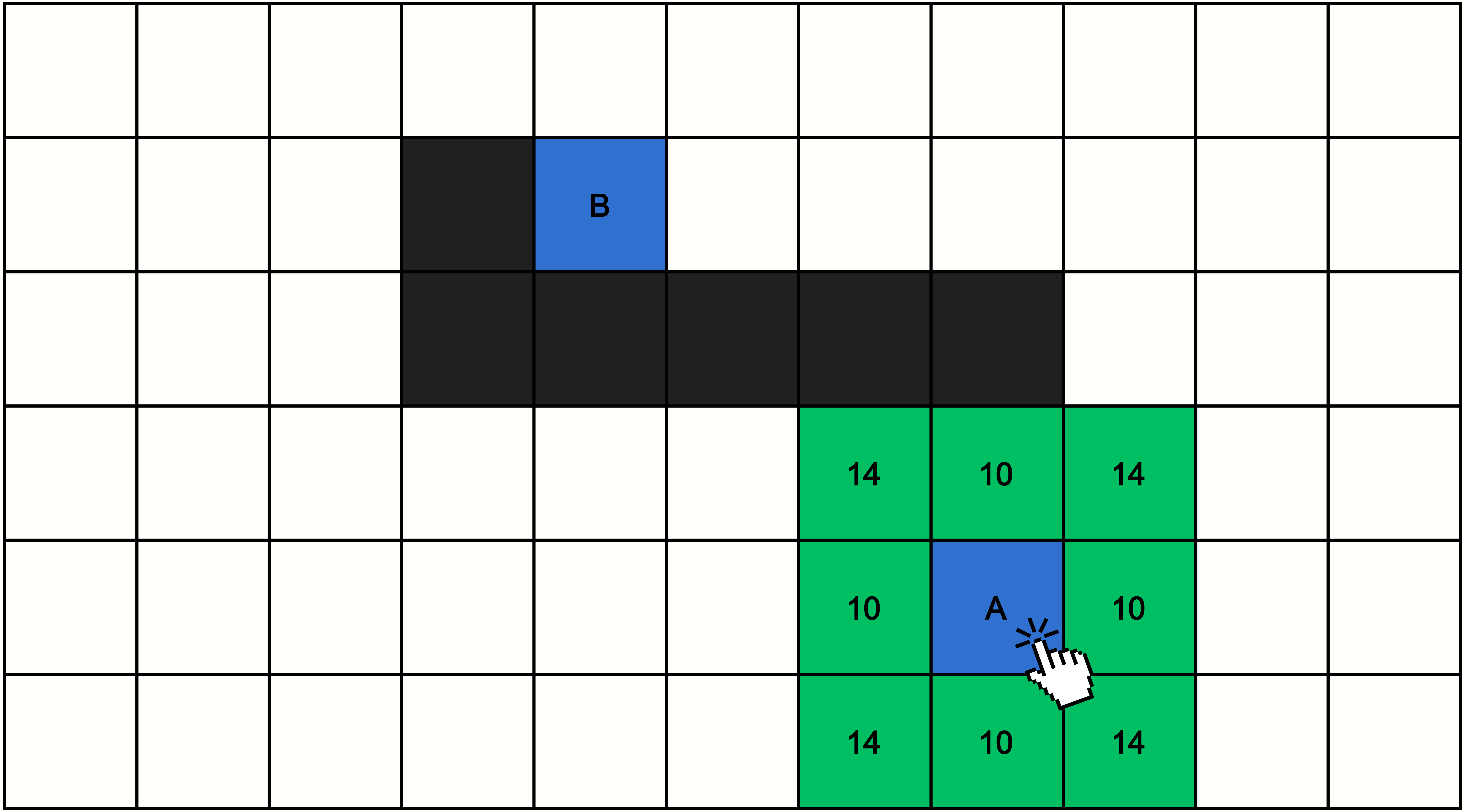
the "look-ahead" effort in searching game trees. Procedures developed via the heuristic approach generally have not been able to guarantee that minimum cost solution paths will always be found.

This paper draws together the above two approaches by describing how information from a problem domain can be incorporated in a formal mathematical approach to a graph analysis problem. It also presents a general algorithm which prescribes how to use such information to find a minimum cost path through a graph. Finally, it proves, under mild assumptions, that this algorithm is optimal in

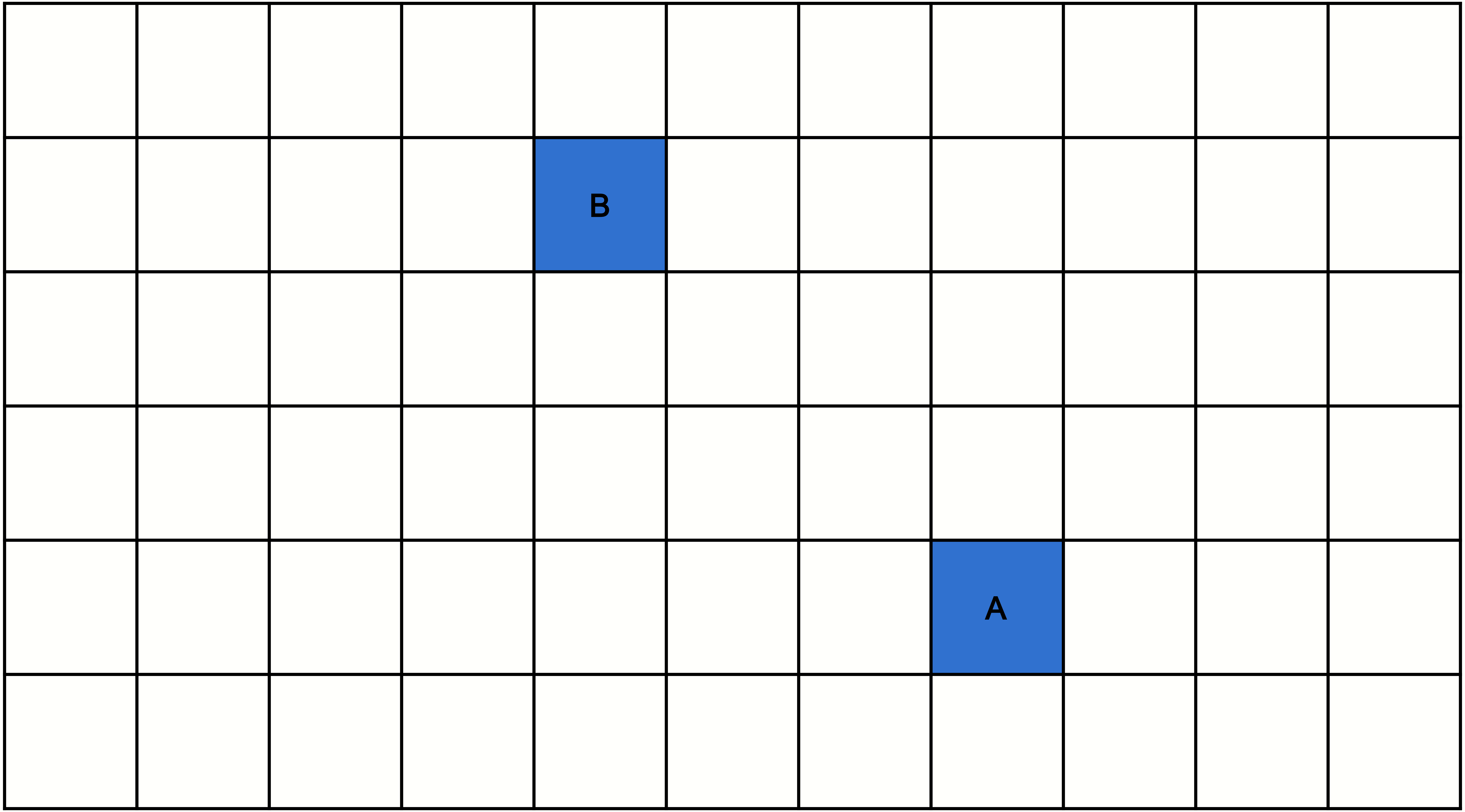
path has a cost which is obtained by adding the individual costs of each arc, $c_{i,i+1}$, in the path. An *optimal path* from n_i to n_j is a path having the smallest cost over the set of all paths from n_i to n_j. We shall represent this cost by $h(n_i, n_j)$.

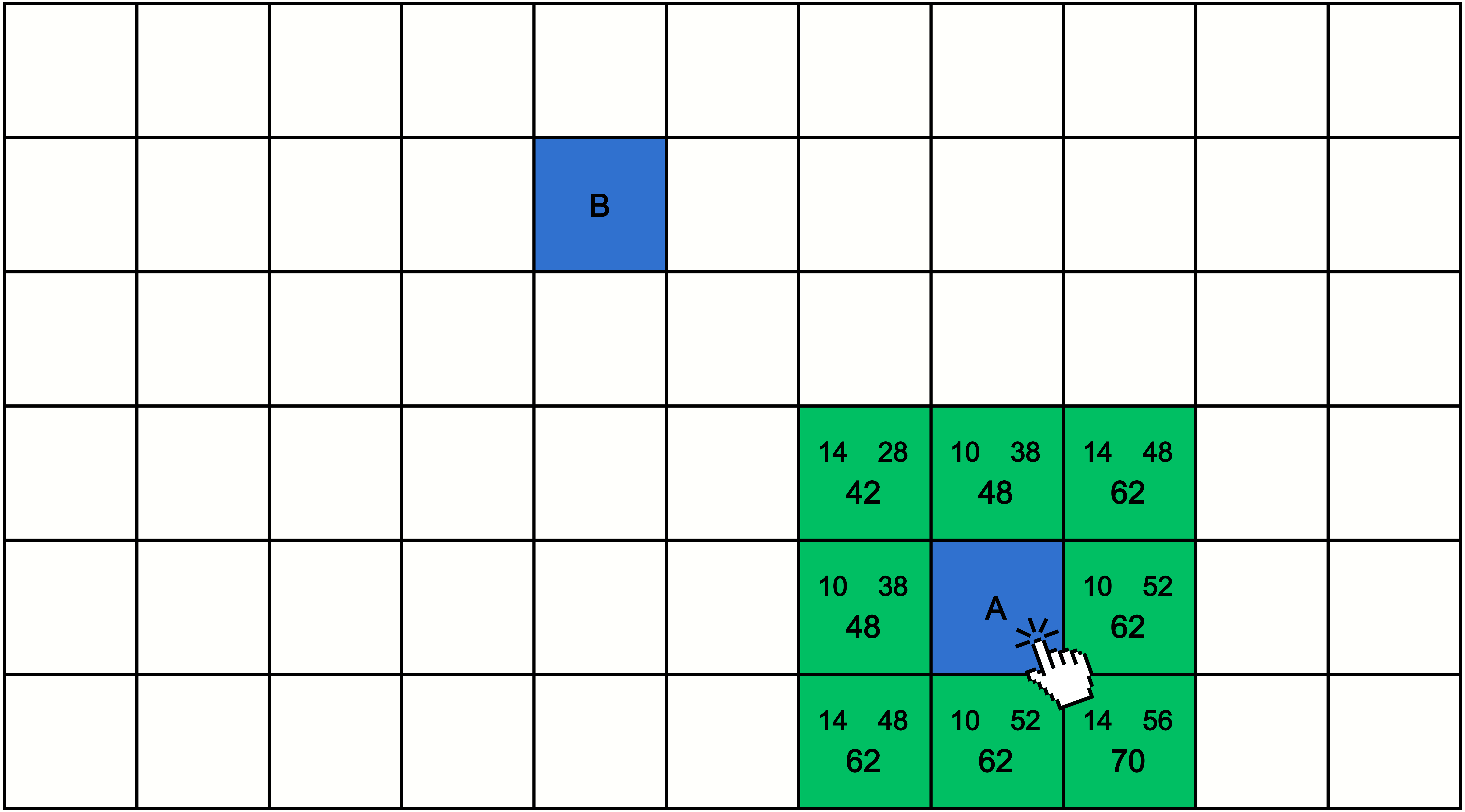
This paper will be concerned with the subgraph G_s from some single specified *start node* s. We define a nonempty set T of nodes in G_s as the *goal nodes.*[1] For any node n in G_s, an element $t \in T$ is a *preferred* goal node of n if and only if the cost of an optimal path from n to t does not exceed the cost of any other path from n to any member of T. For simplicity, we shall represent the unique cost of an

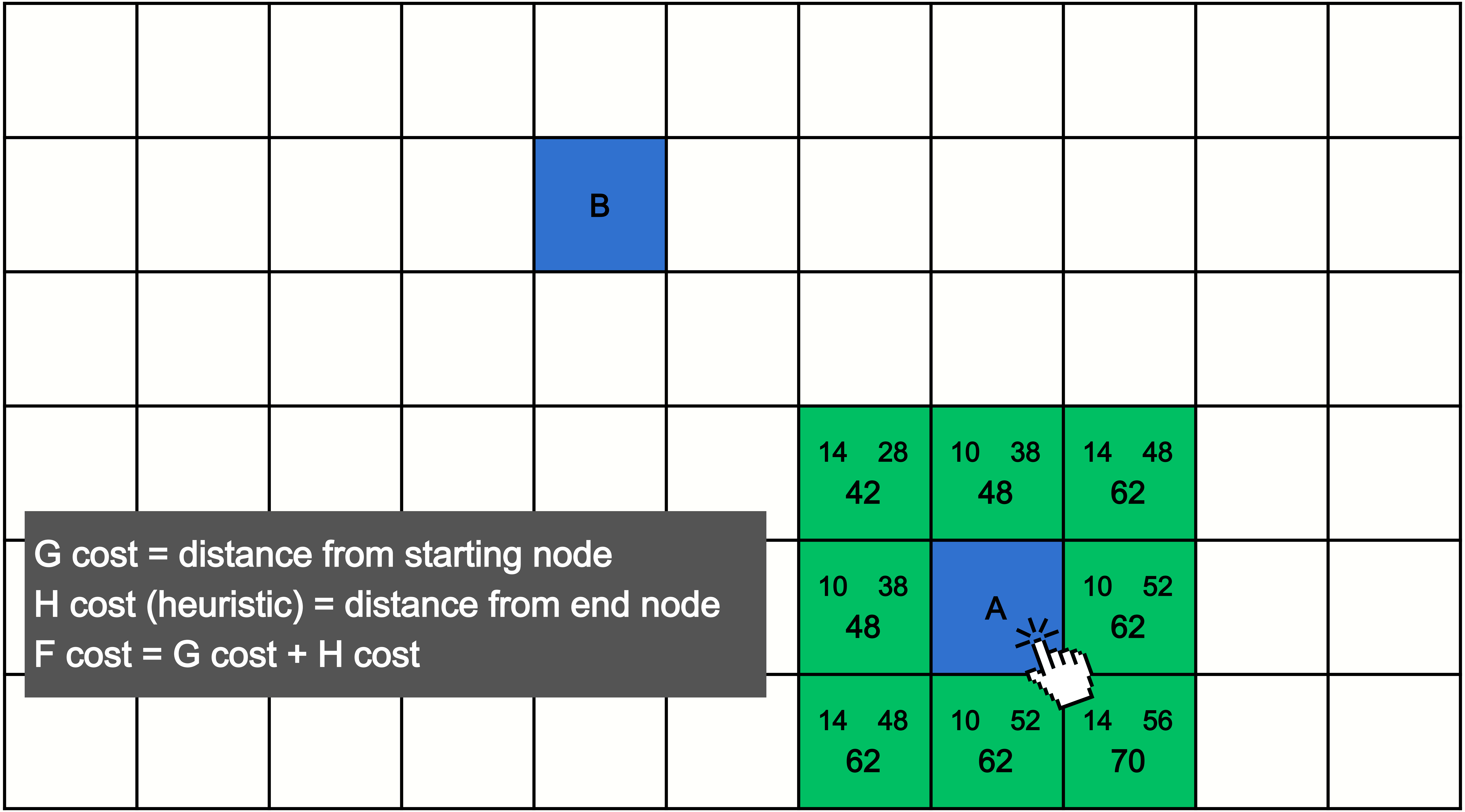

B
A

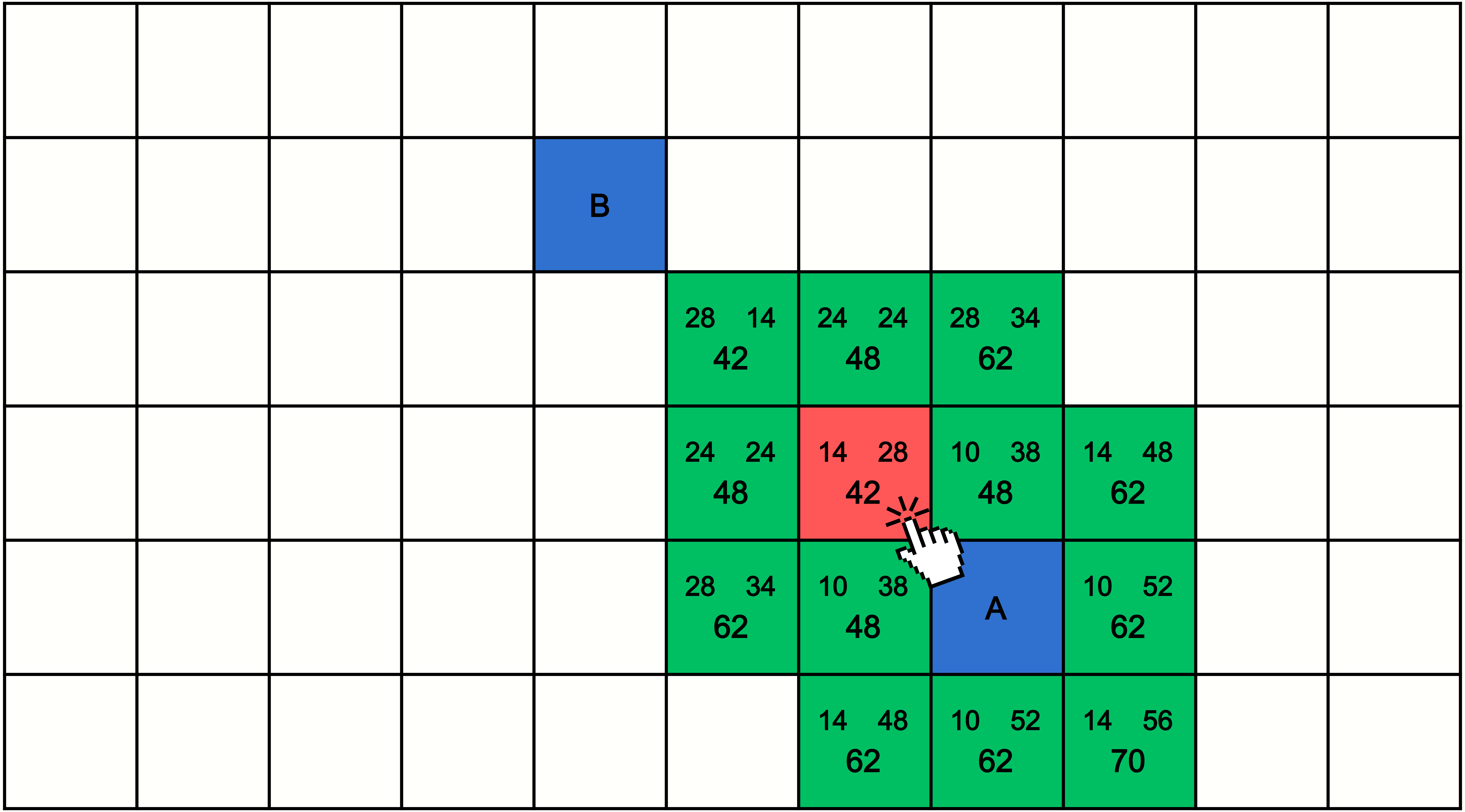


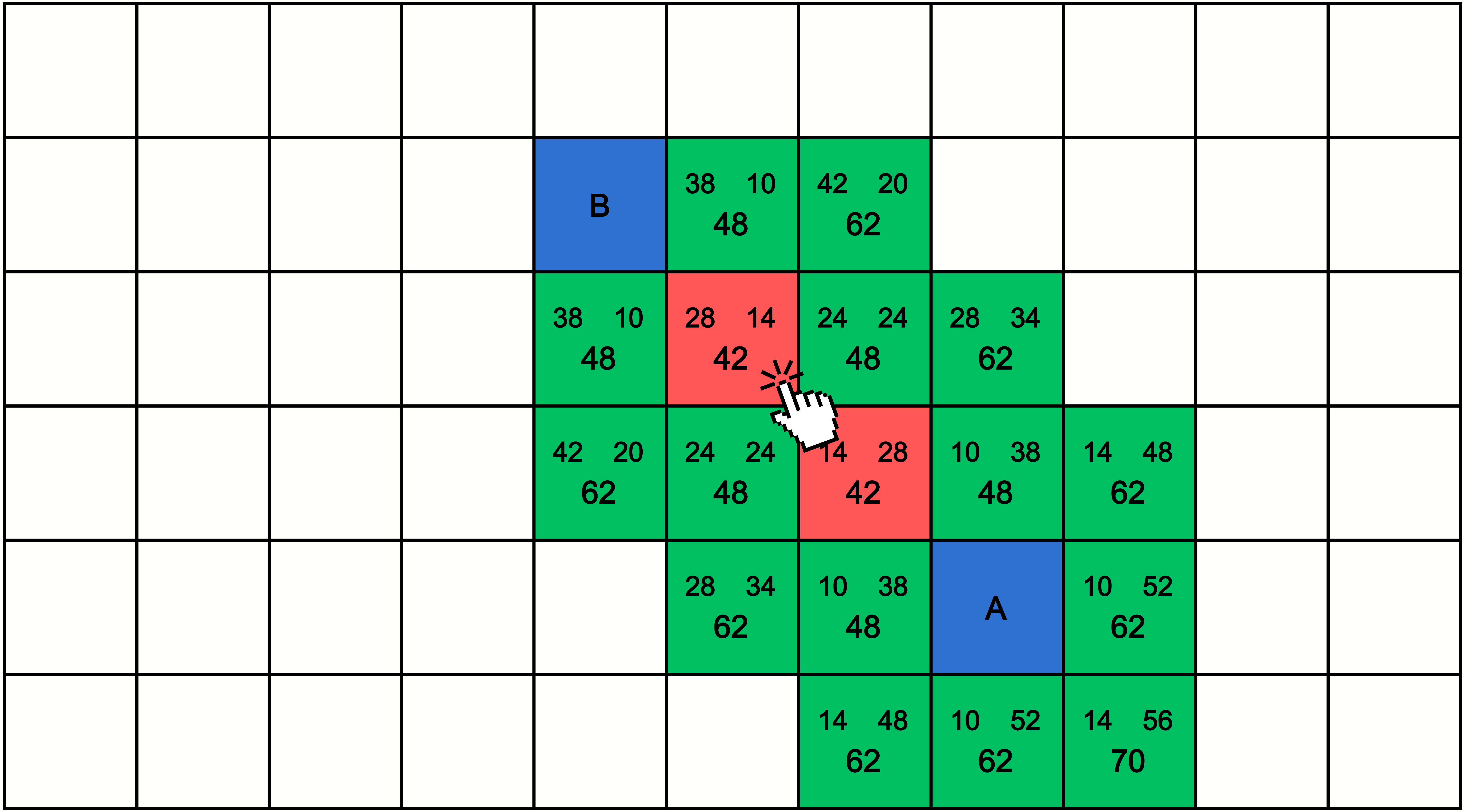
B
14
10
14
10
A
10
14
10
14


B
A

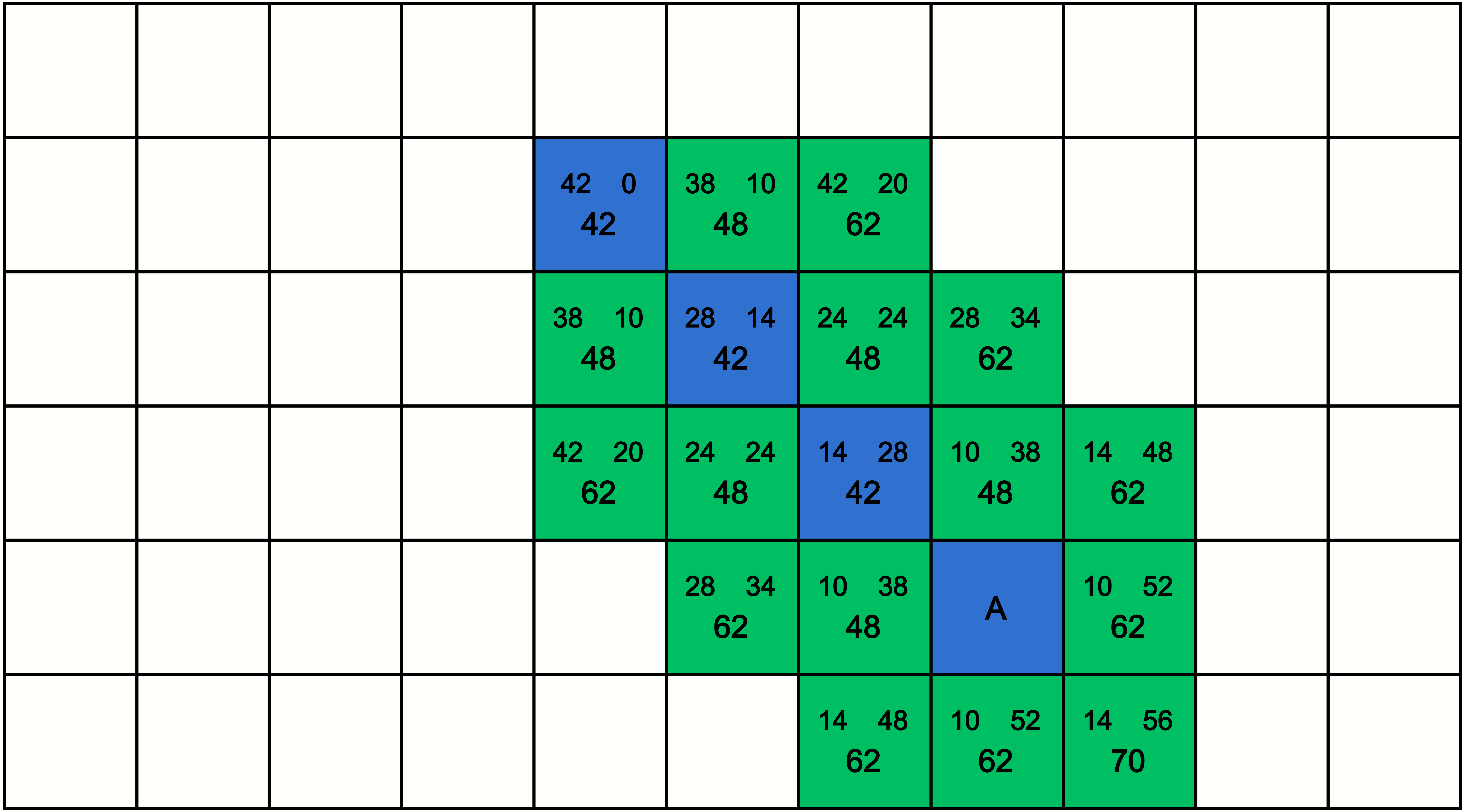

B
14 28
42
10 38
48
14 48
62
10 38
48
A
10 52
62
14 48
62
10 52
62
14 56
70

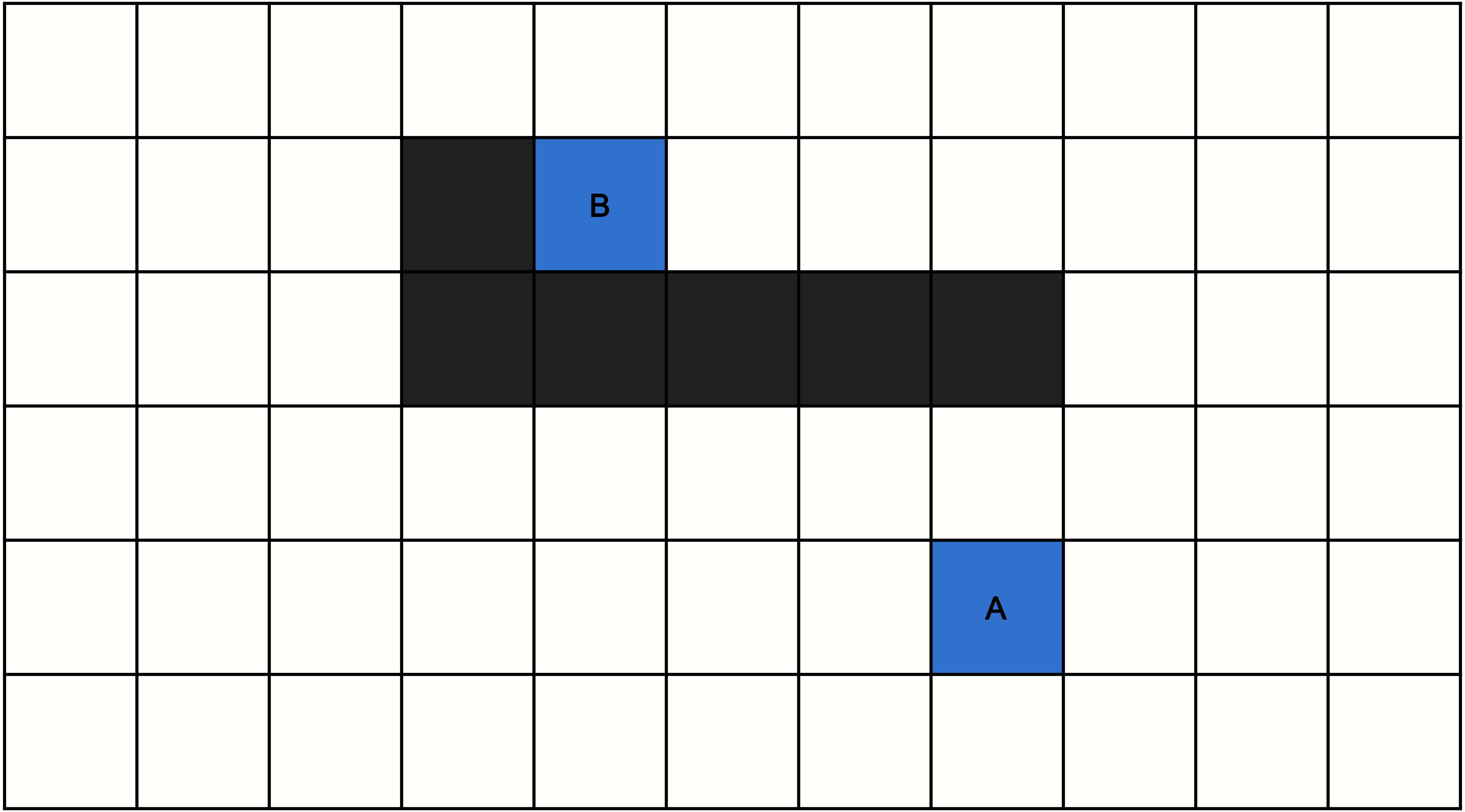

B
14 28
42
10 38
48
14 48
62
10 38
48
A
10 52
62
14 48
62
10 52
62
14 56
70
G cost = distance from starting node
H cost (heuristic) = distance from end node
F cost = G cost + H cost

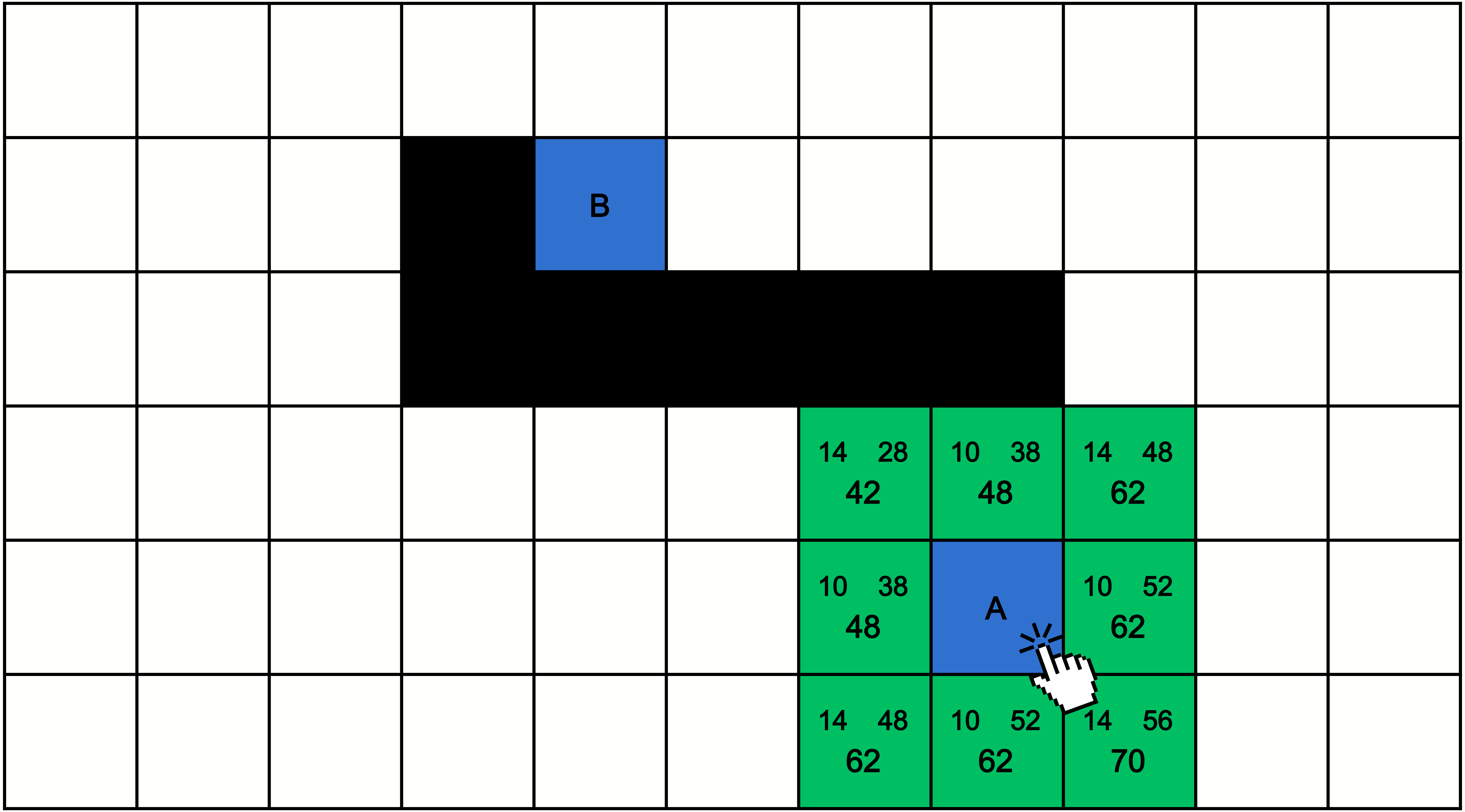

B
28 14 42
24 24 48
28 34 62
24 24 48
14 28 42
10 38 48
14 48 62
28 34 62
10 38 48
A
10 52 62
14 48 62
10 52 62
14 56 70

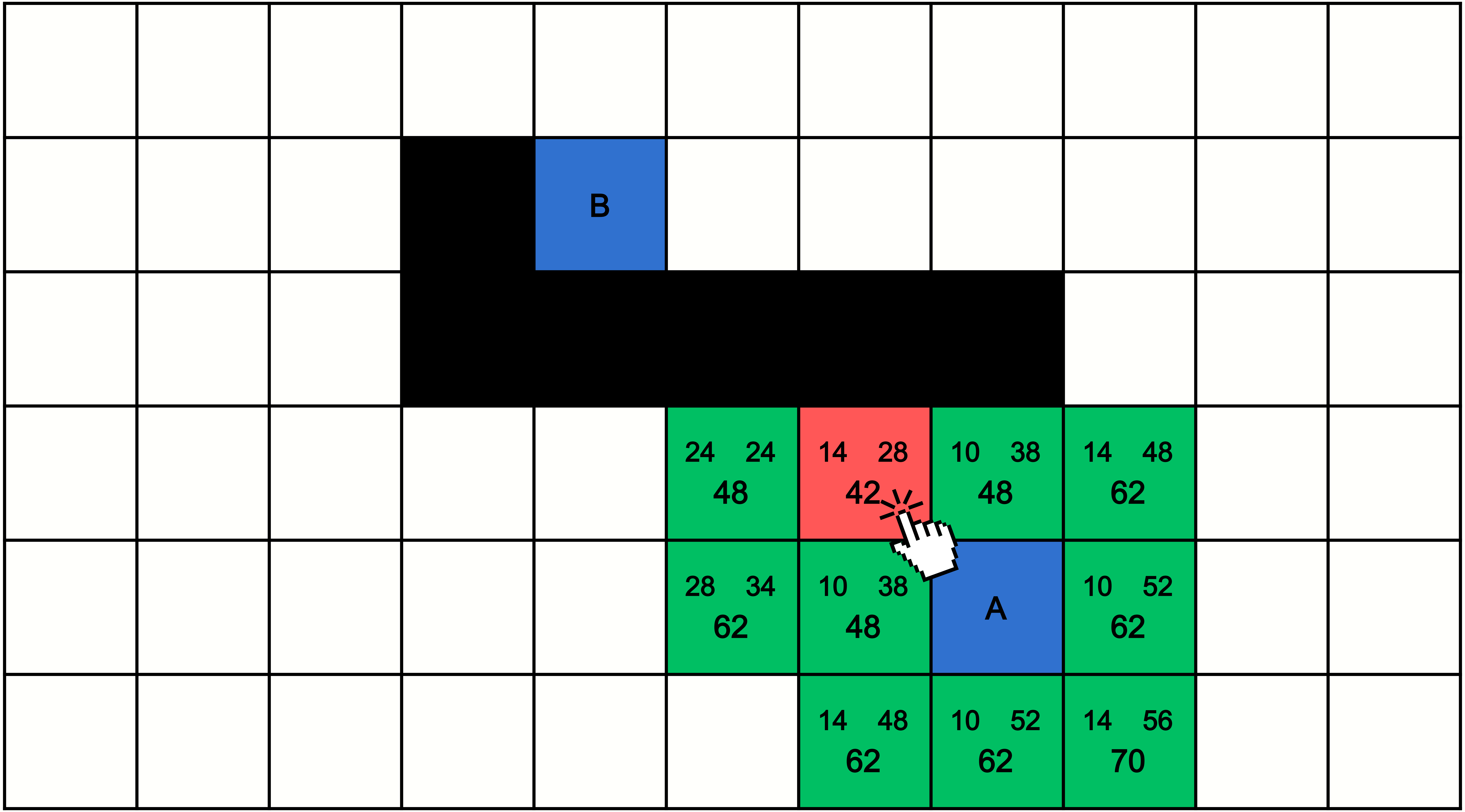


B
38 10 48
42 20 62
38 10 48
28 14 42
24 24 48
28 34 62
42 20 62
24 24 48
14 28 42
10 38 48
14 48 62
28 34 62
10 38 48
A
10 52 62
14 48 62
10 52 62
14 56 70

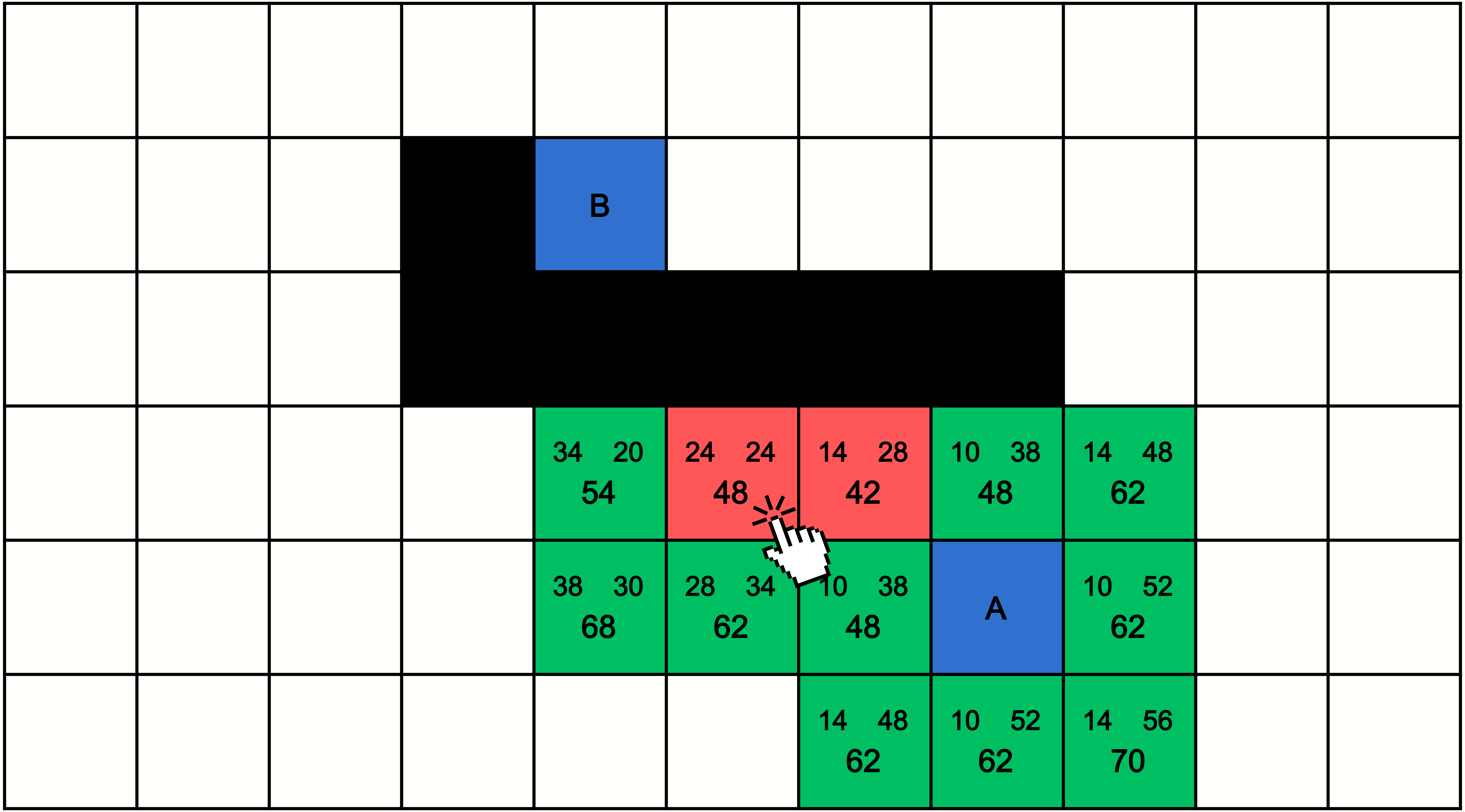

42 0
42
38 10
48
42 20
62
38 10
48
28 14
42
24 24
48
28 34
62
42 20
62
24 24
48
14 28
42
10 38
48
14 48
62
28 34
62
10 38
48
A
10 52
62
14 48
62
10 52
62
14 56
70

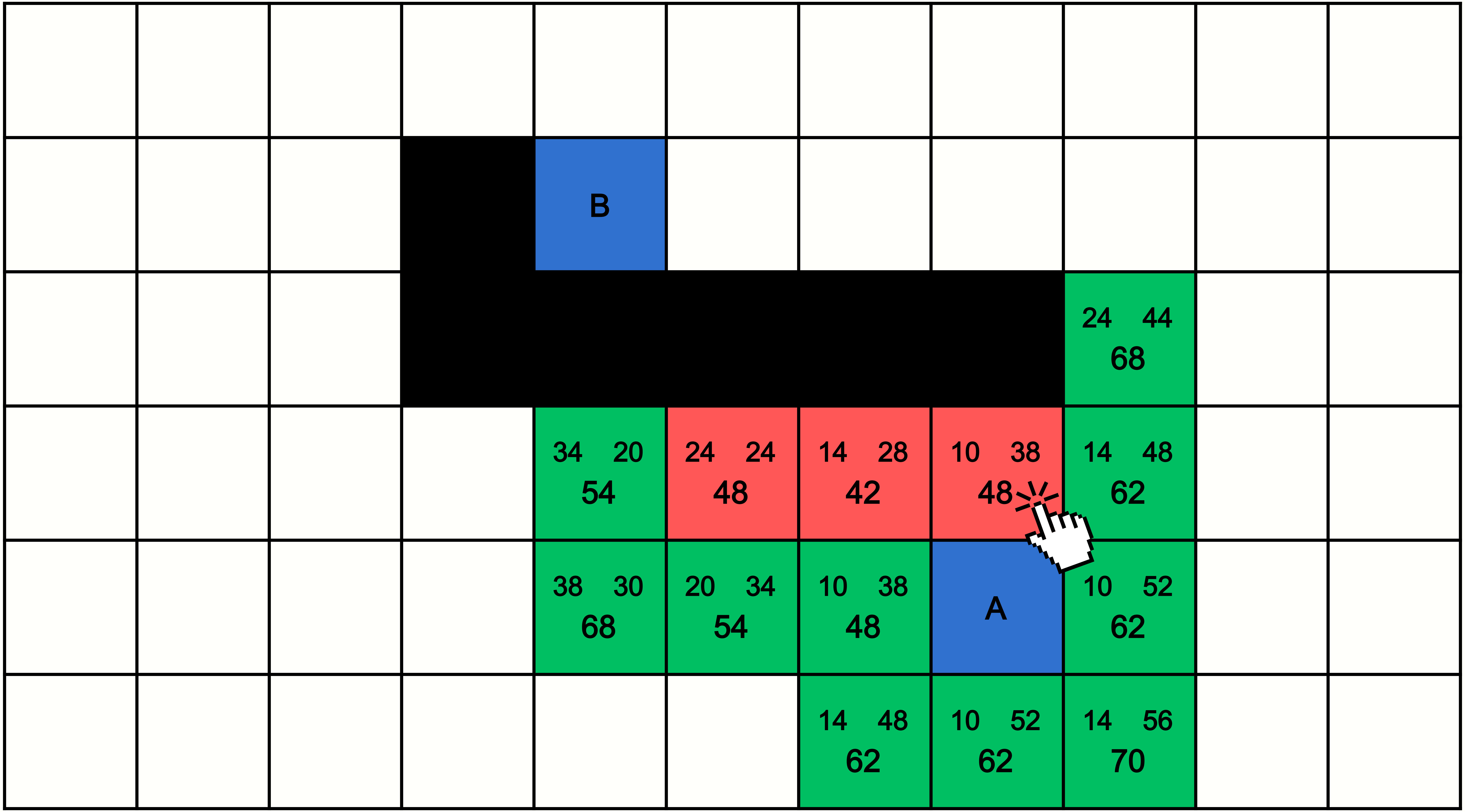

B
A

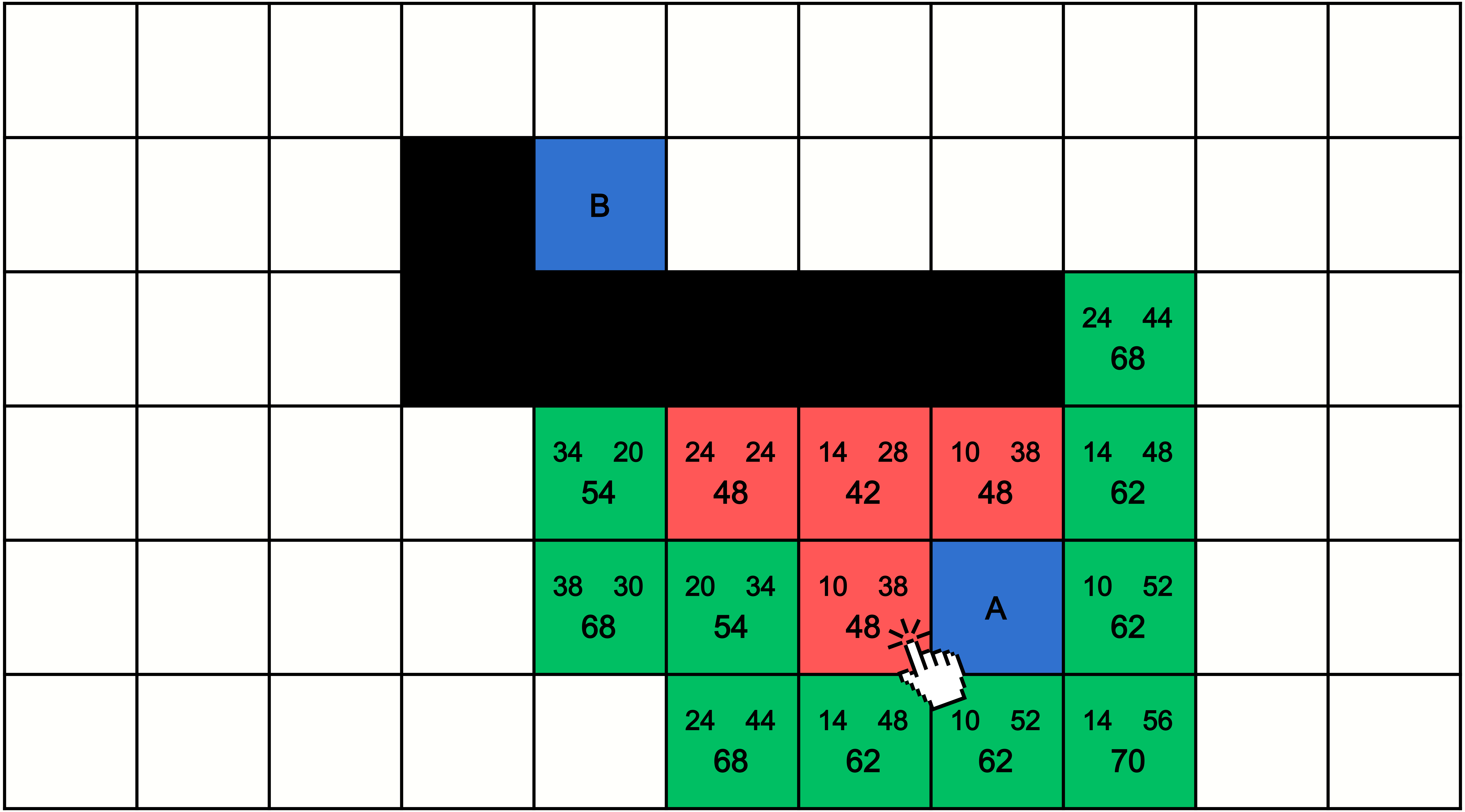

B
14 28
42
10 38
48
14 48
62
10 38
48
A
10 52
62
14 48
62
10 52
62
14 56
70

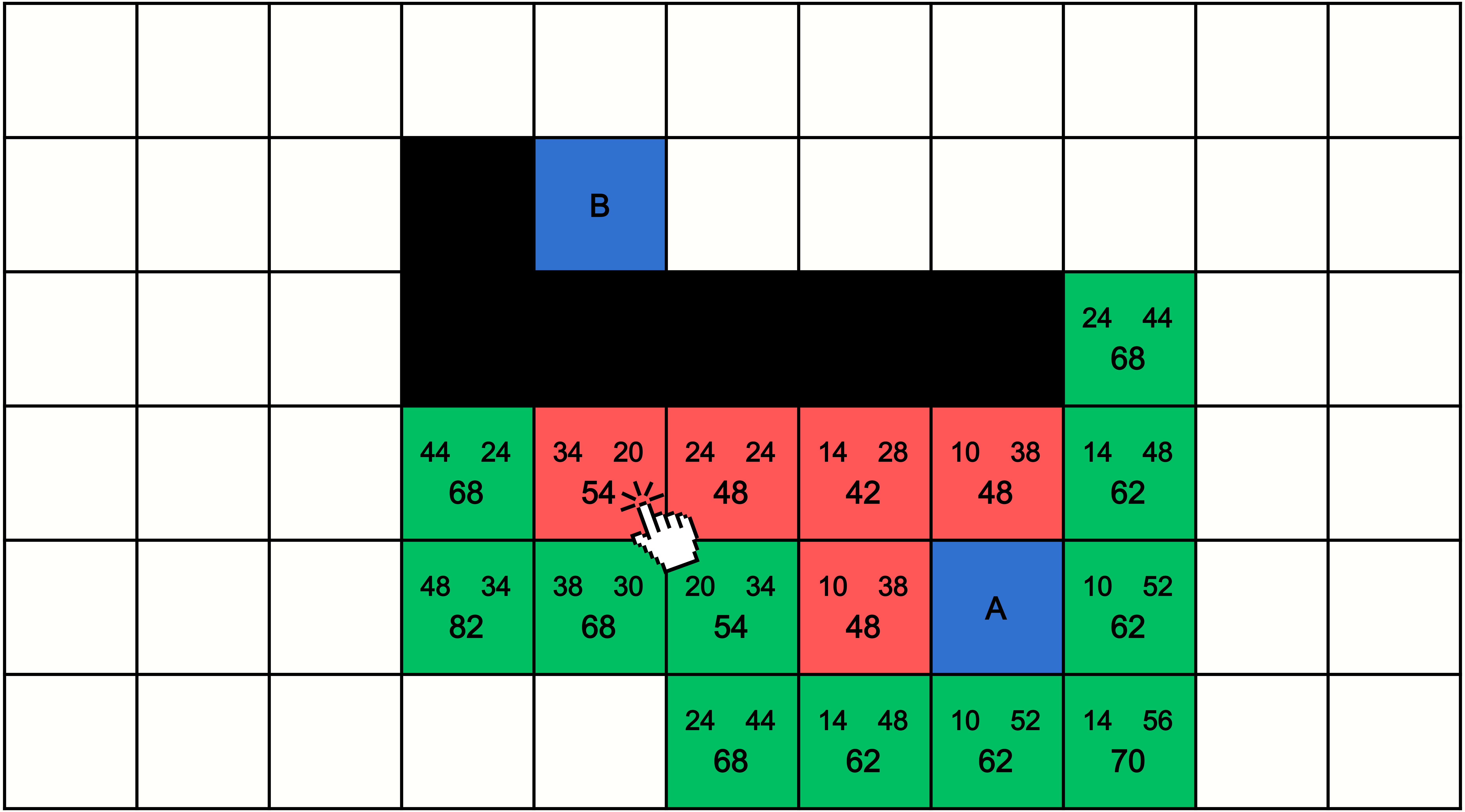


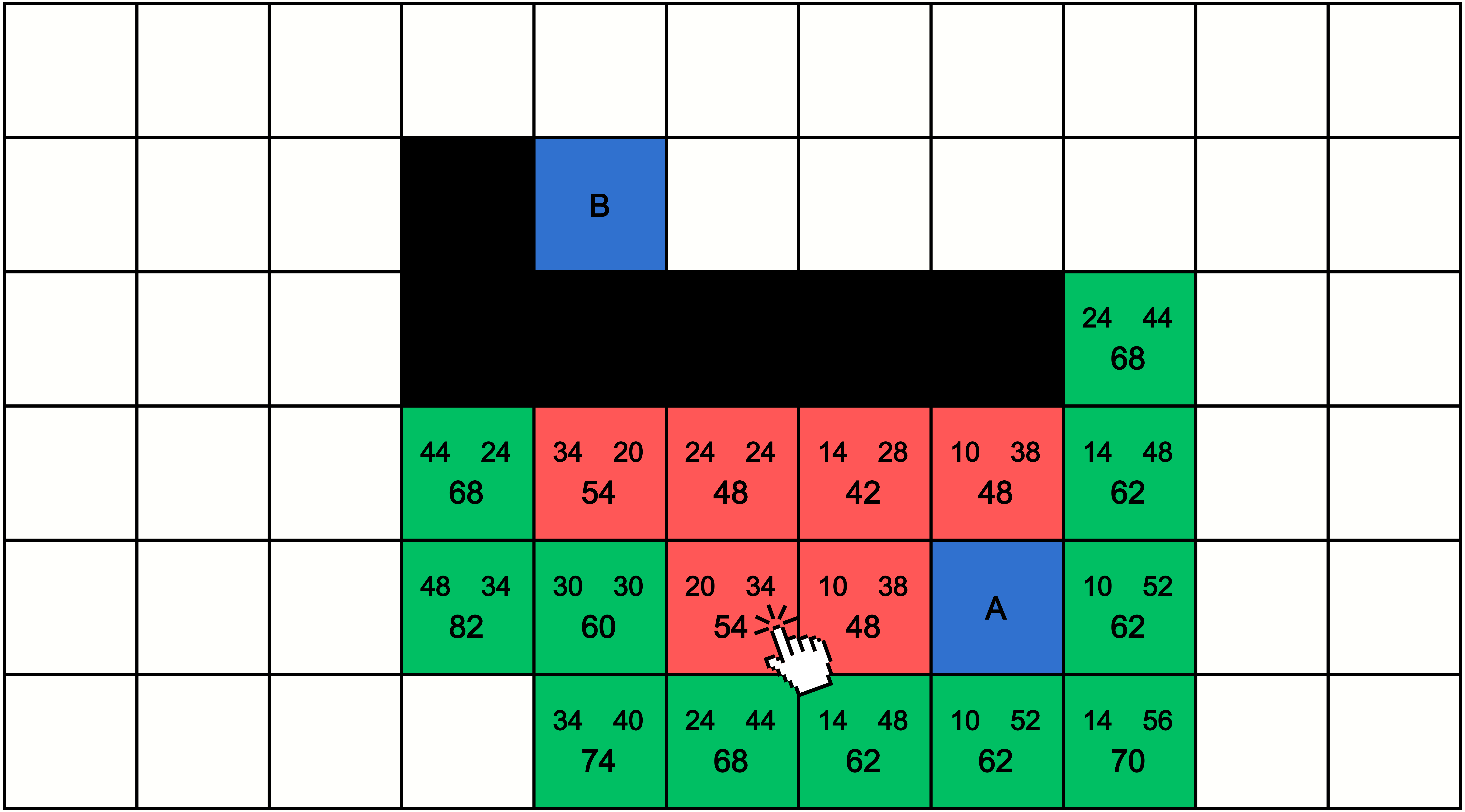
B
24 24 48
14 28 42
10 38 48
14 48 62
28 34 62
10 38 48
A
10 52 62
14 48 62
10 52 62
14 56 70

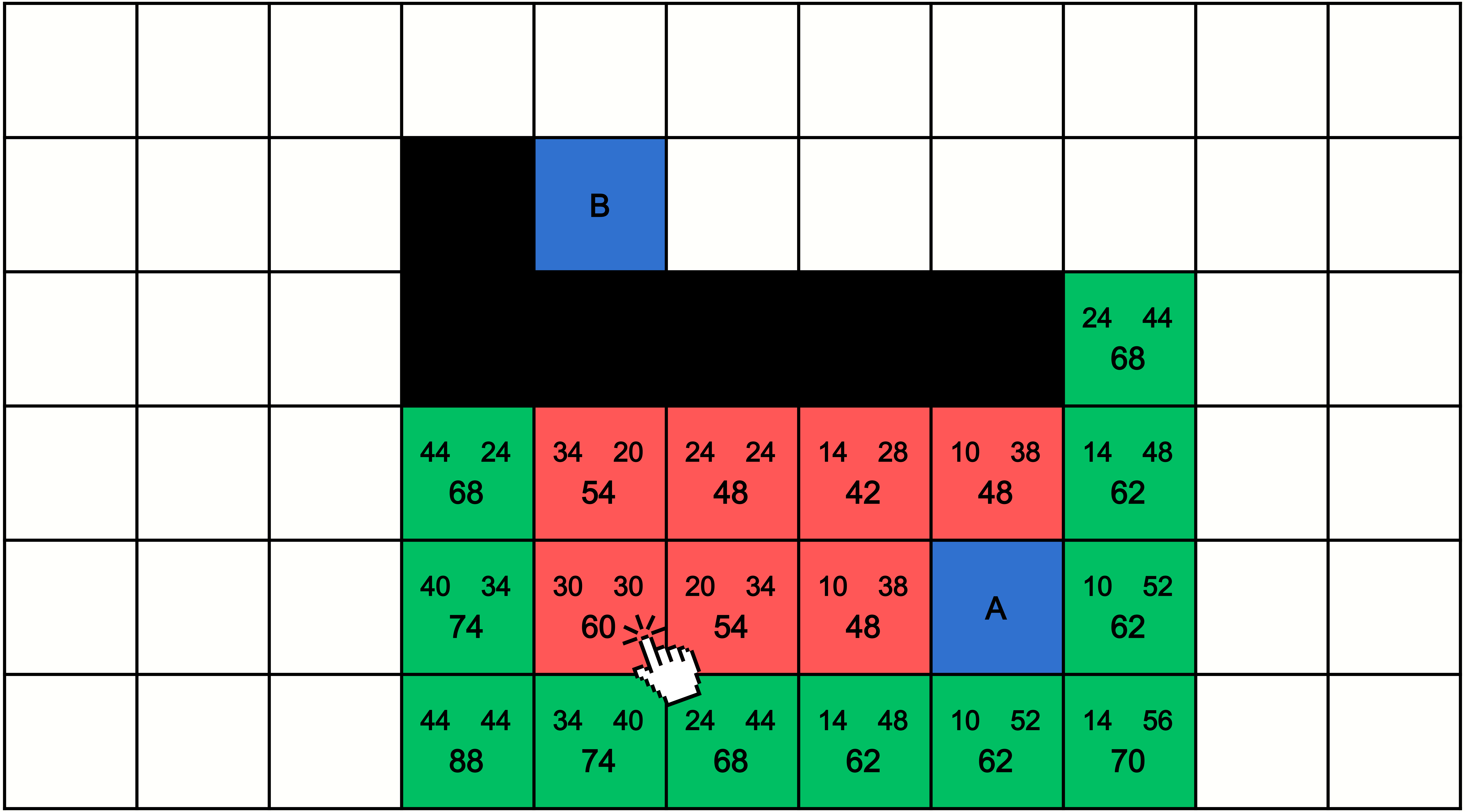

B
34 20 54
24 24 48
14 28 42
10 38 48
14 48 62
38 30 68
28 34 62
10 38 48
A
10 52 62
14 48 62
10 52 62
14 56 70

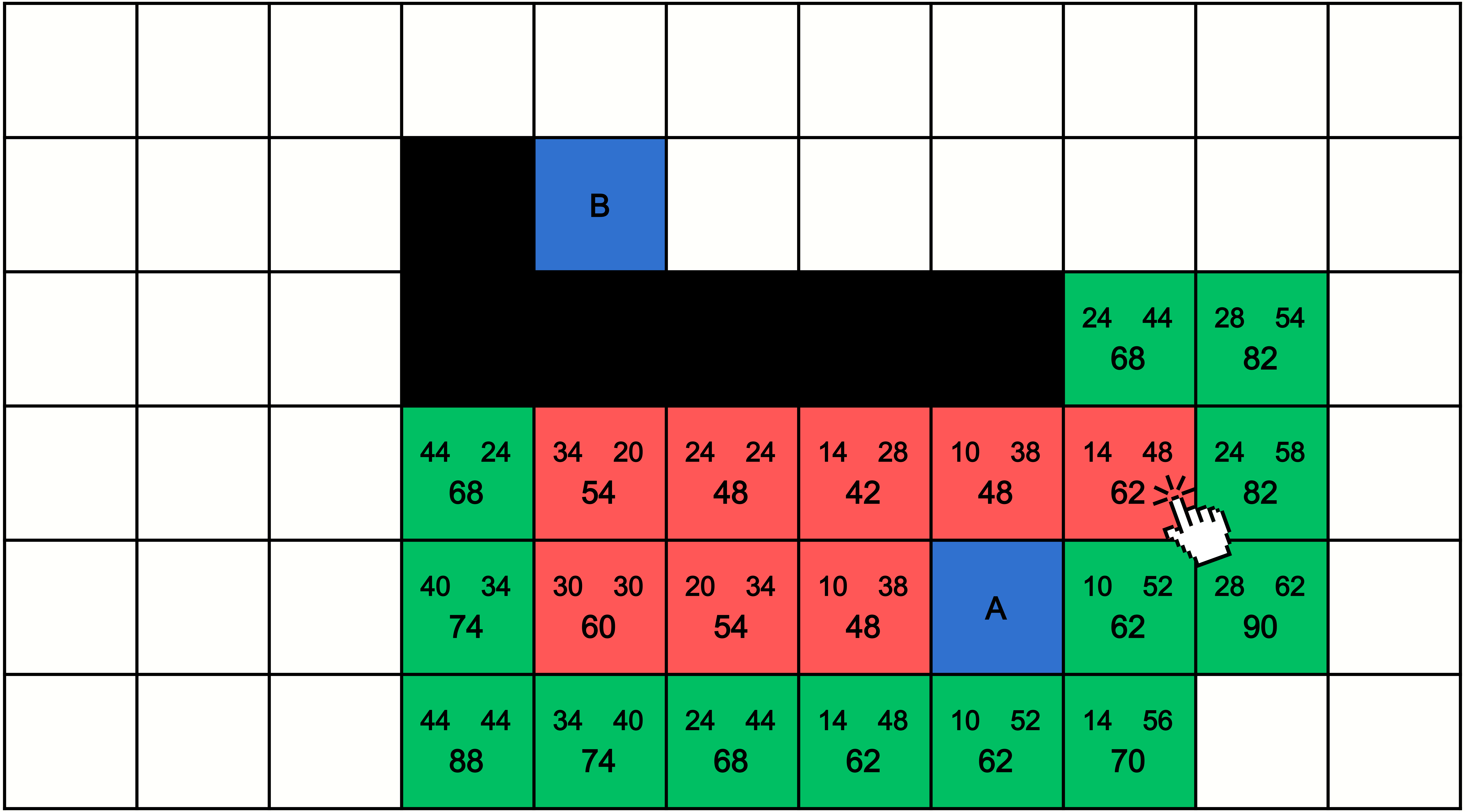

B
24 44
68
34 20
54
24 24
48
14 28
42
10 38
48
14 48
62
38 30
68
20 34
54
10 38
48
A
10 52
62
14 48
62
10 52
62
14 56
70

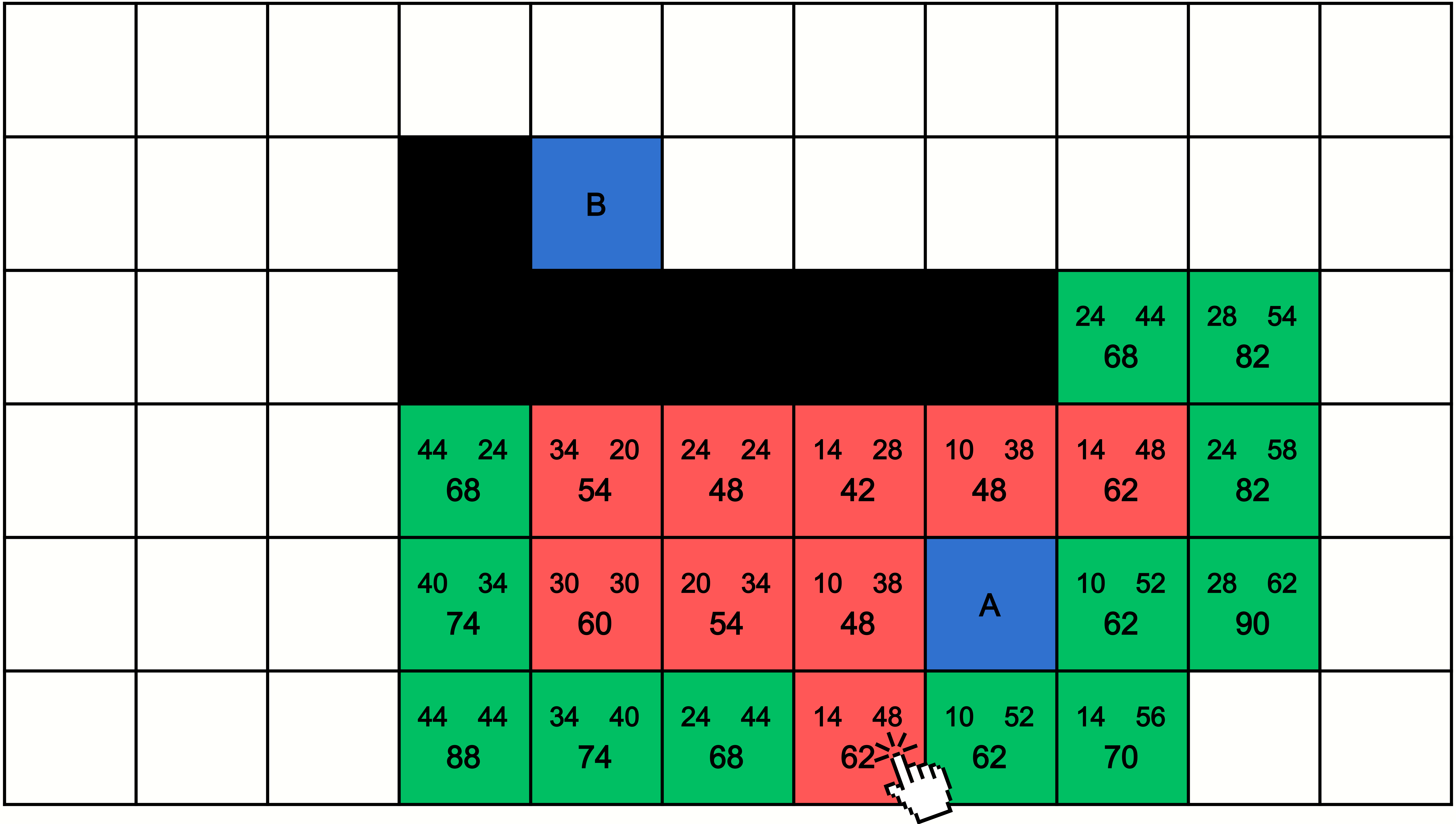

B
24 44 68
34 20 54
24 24 48
14 28 42
10 38 48
14 48 62
38 30 68
20 34 54
10 38 48
A
10 52 62
24 44 68
14 48 62
10 52 62
14 56 70

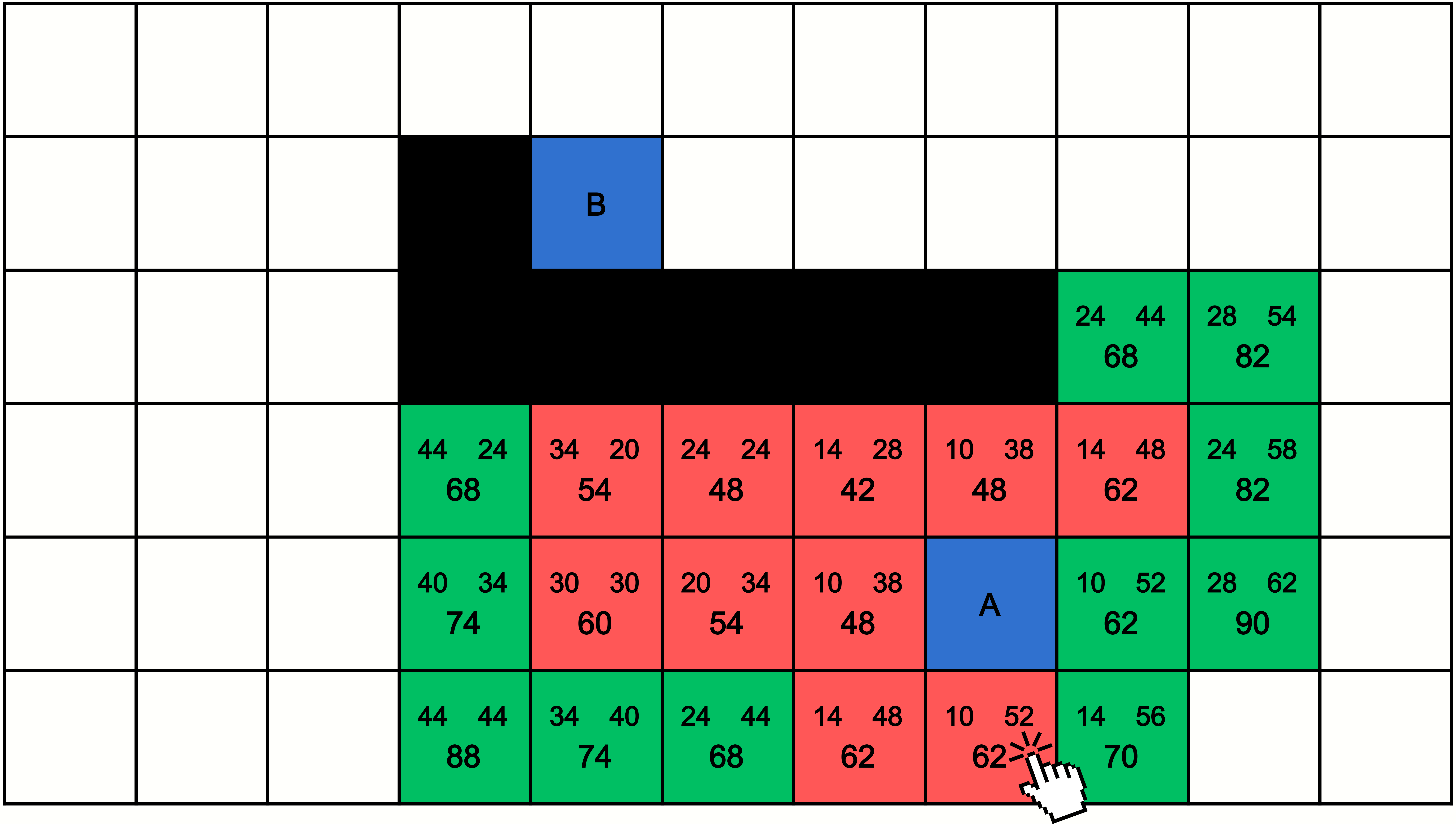

B
24 44
68
44 24
68
34 20
54
24 24
48
14 28
42
10 38
48
14 48
62
48 34
82
38 30
68
20 34
54
10 38
48
A
10 52
62
24 44
68
14 48
62
10 52
62
14 56
70

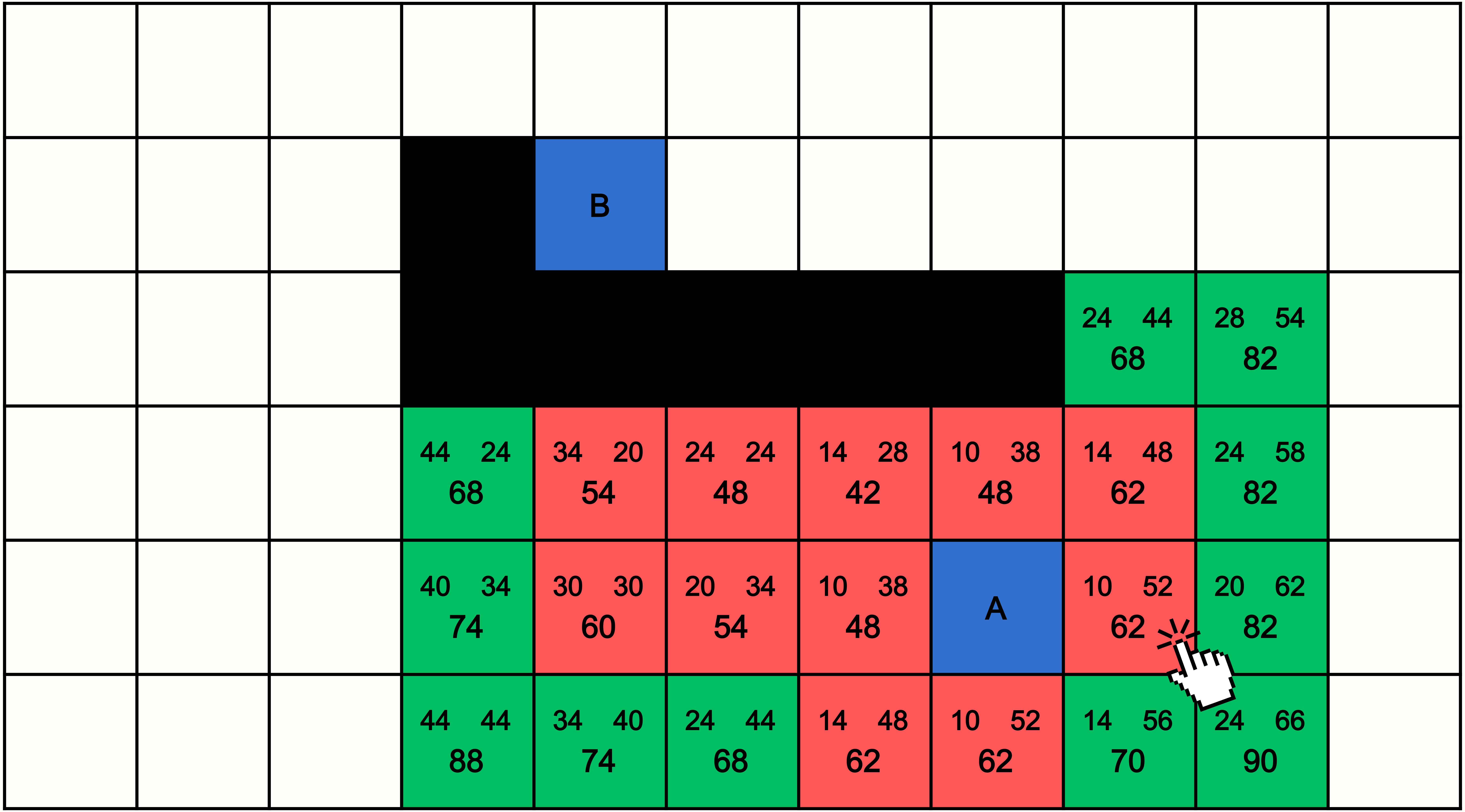

B
24 44 68
44 24 68
34 20 54
24 24 48
14 28 42
10 38 48
14 48 62
48 34 82
30 30 60
20 34 54
10 38 48
A
10 52 62
34 40 74
24 44 68
14 48 62
10 52 62
14 56 70

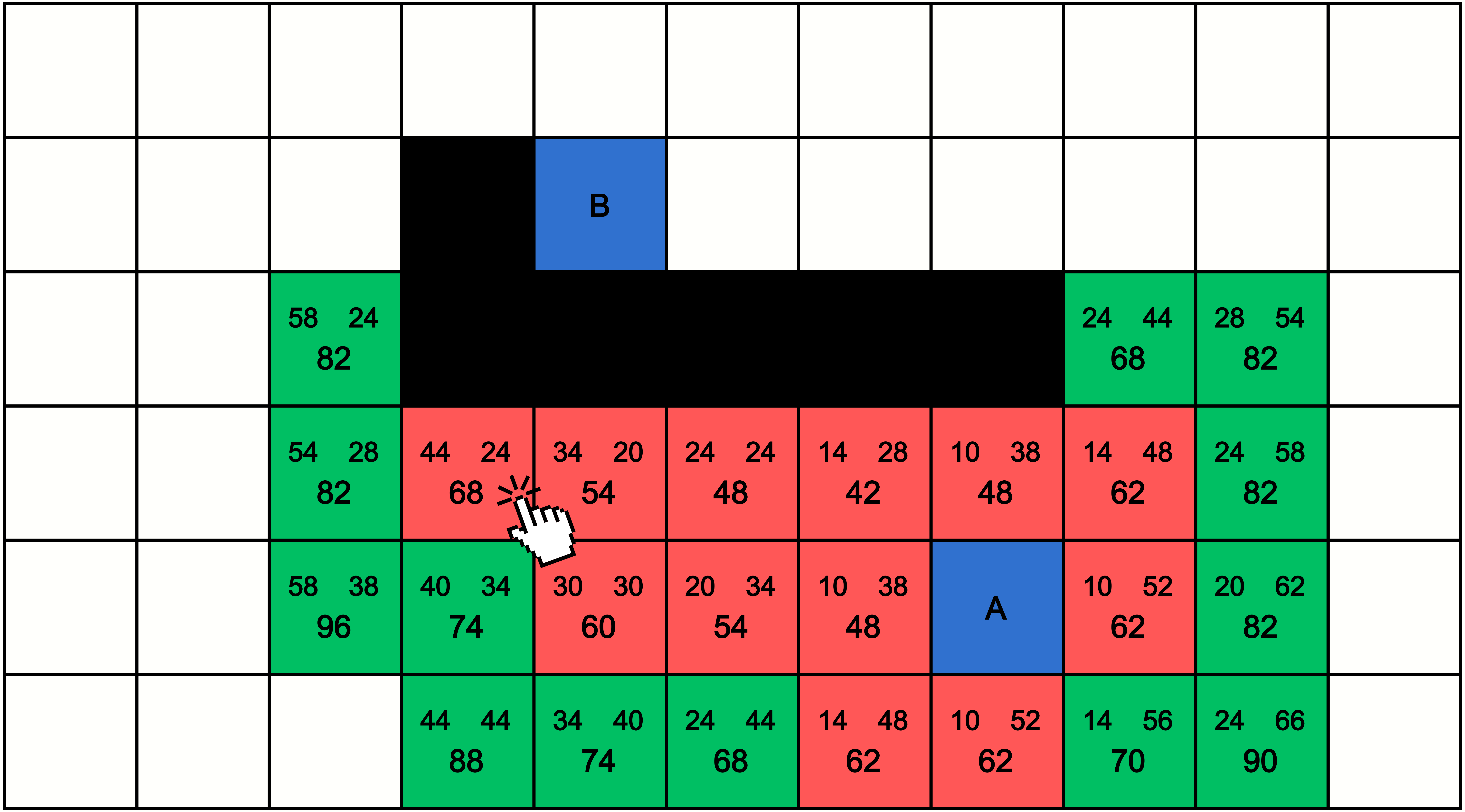

B
24 44 68
44 24 68
34 20 54
24 24 48
14 28 42
10 38 48
14 48 62
40 34 74
30 30 60
20 34 54
10 38 48
A
10 52 62
44 44 88
34 40 74
24 44 68
14 48 62
10 52 62
14 56 70


B
24 44 68
28 54 82
44 24 68
34 20 54
24 24 48
14 28 42
10 38 48
14 48 62
24 58 82
40 34 74
30 30 60
20 34 54
10 38 48
A
10 52 62
28 62 90
44 44 88
34 40 74
24 44 68
14 48 62
10 52 62
14 56 70

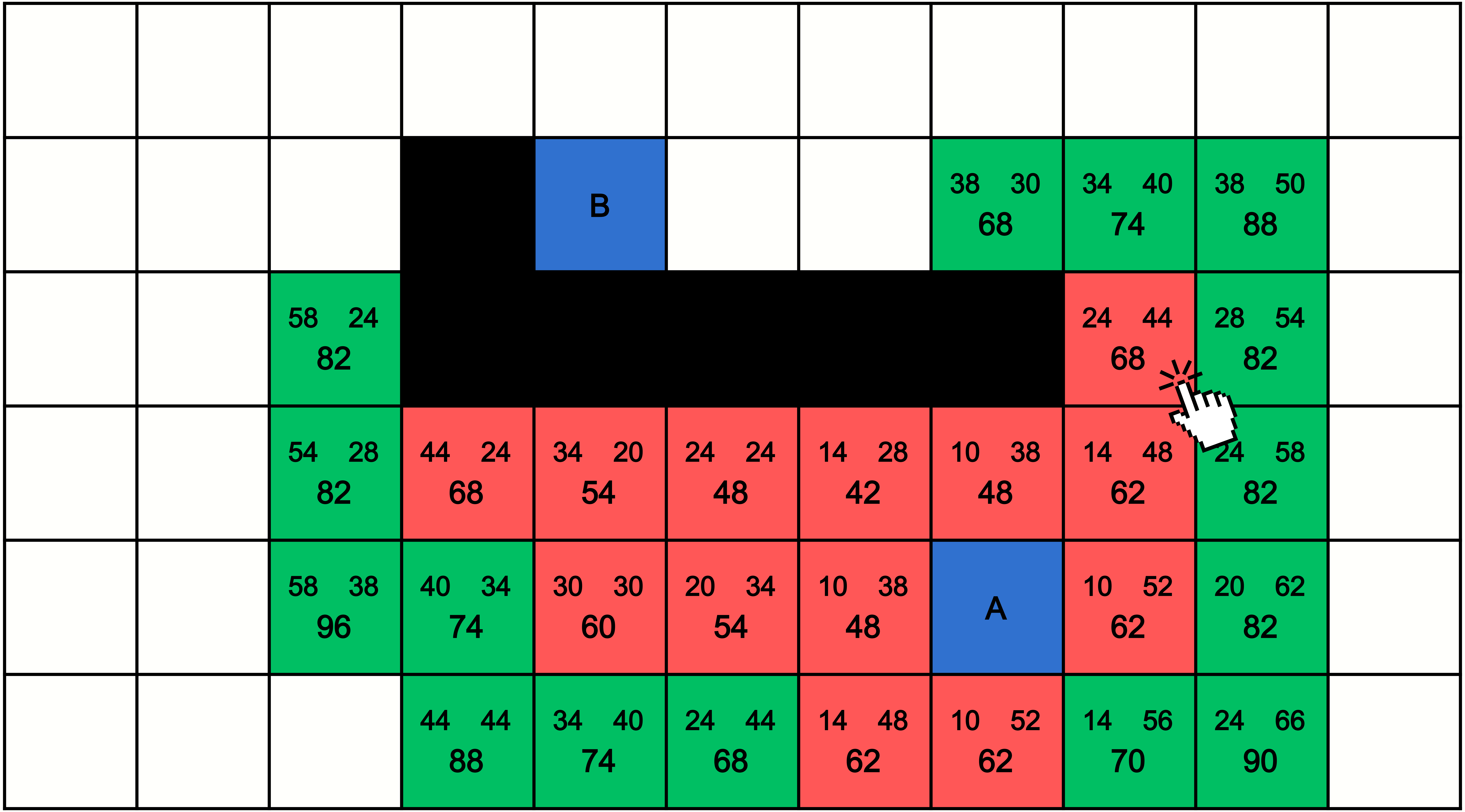

B
24 44 68
28 54 82
44 24 68
34 20 54
24 24 48
14 28 42
10 38 48
14 48 62
24 58 82
40 34 74
30 30 60
20 34 54
10 38 48
A
10 52 62
28 62 90
44 44 88
34 40 74
24 44 68
14 48 62
10 52 62
14 56 70

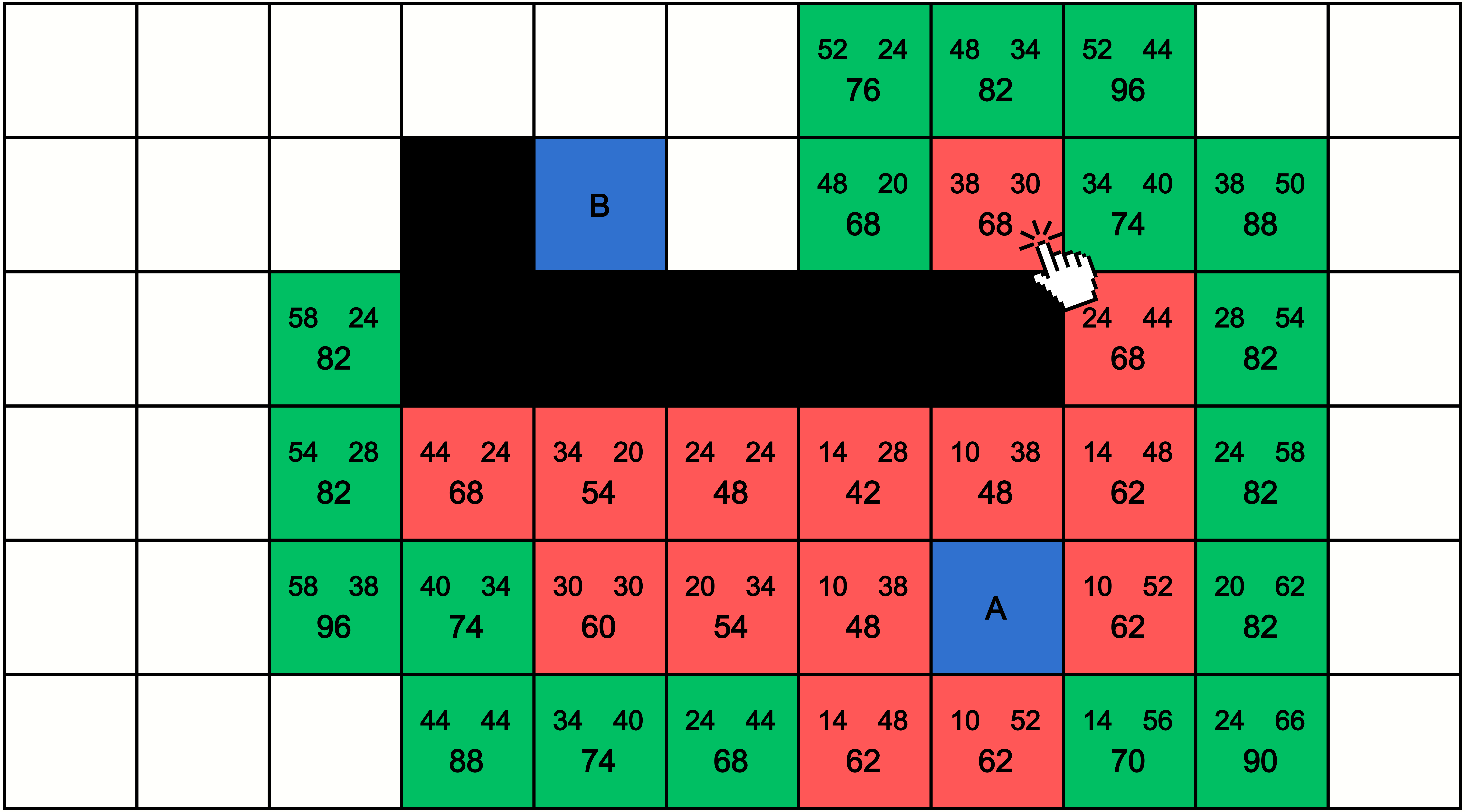

B
24 44
68
28 54
82
44 24
68
34 20
54
24 24
48
14 28
42
10 38
48
14 48
62
24 58
82
40 34
74
30 30
60
20 34
54
10 38
48
A
10 52
62
28 62
90
44 44
88
34 40
74
24 44
68
14 48
62
10 52
62
14 56
70

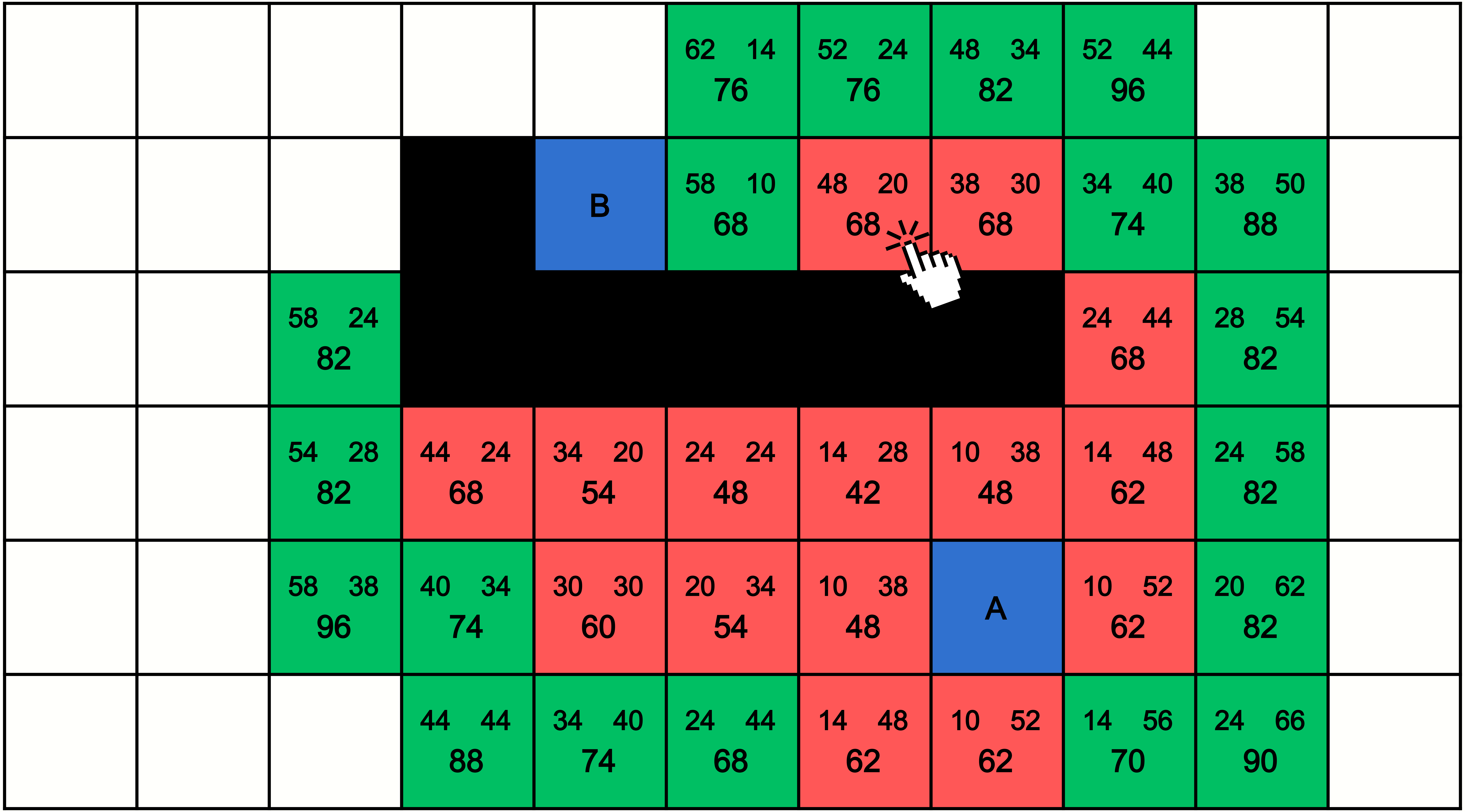

B
24 44 68
28 54 82
44 24 68
34 20 54
24 24 48
14 28 42
10 38 48
14 48 62
24 58 82
40 34 74
30 30 60
20 34 54
10 38 48
A
10 52 62
20 62 82
44 44 88
34 40 74
24 44 68
14 48 62
10 52 62
14 56 70
24 66 90

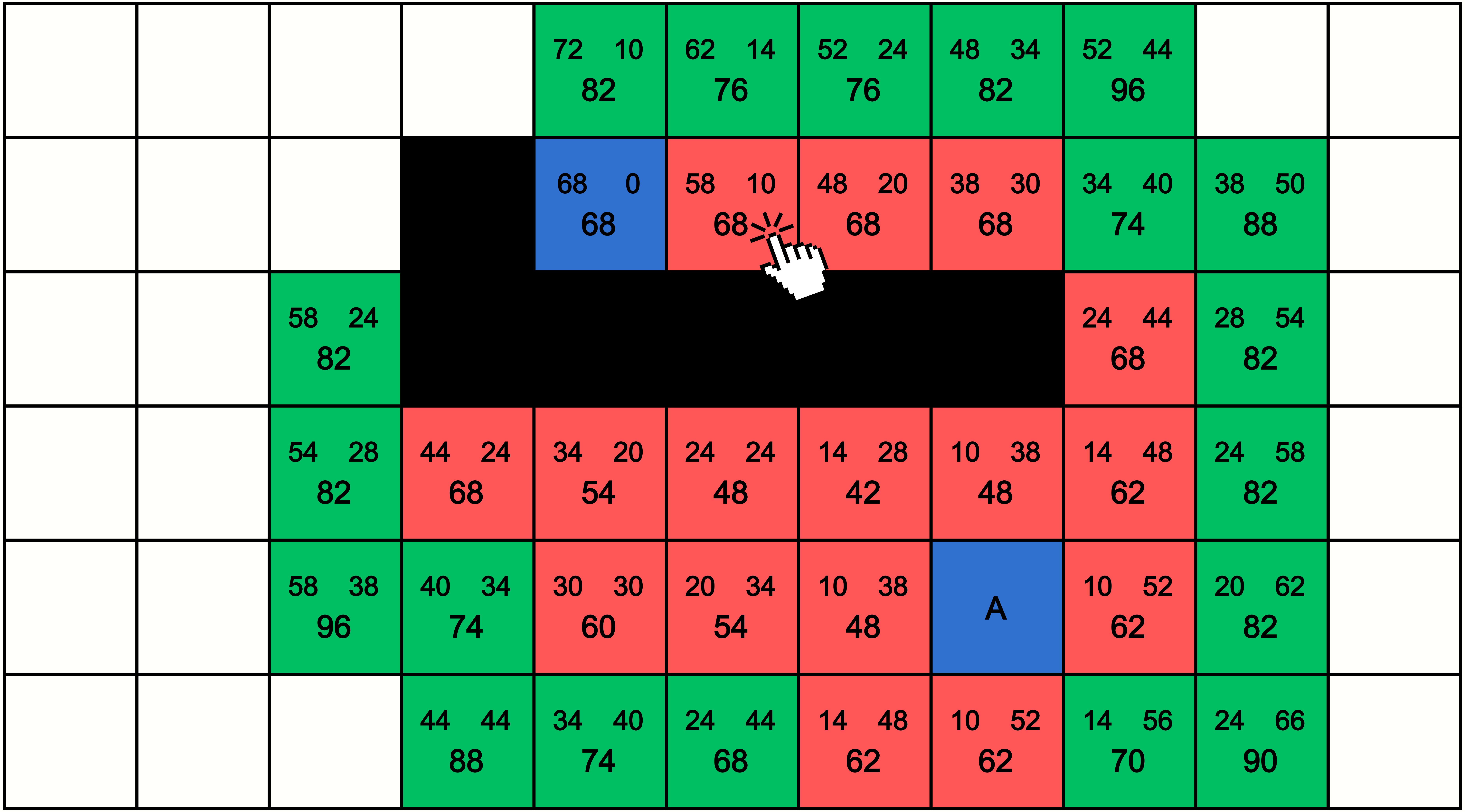


B
58 24 82
24 44 68
28 54 82
54 28 82
44 24 68
34 20 54
24 24 48
14 28 42
10 38 48
14 48 62
24 58 82
58 38 96
40 34 74
30 30 60
20 34 54
10 38 48
A
10 52 62
20 62 82
44 44 88
34 40 74
24 44 68
14 48 62
10 52 62
14 56 70
24 66 90

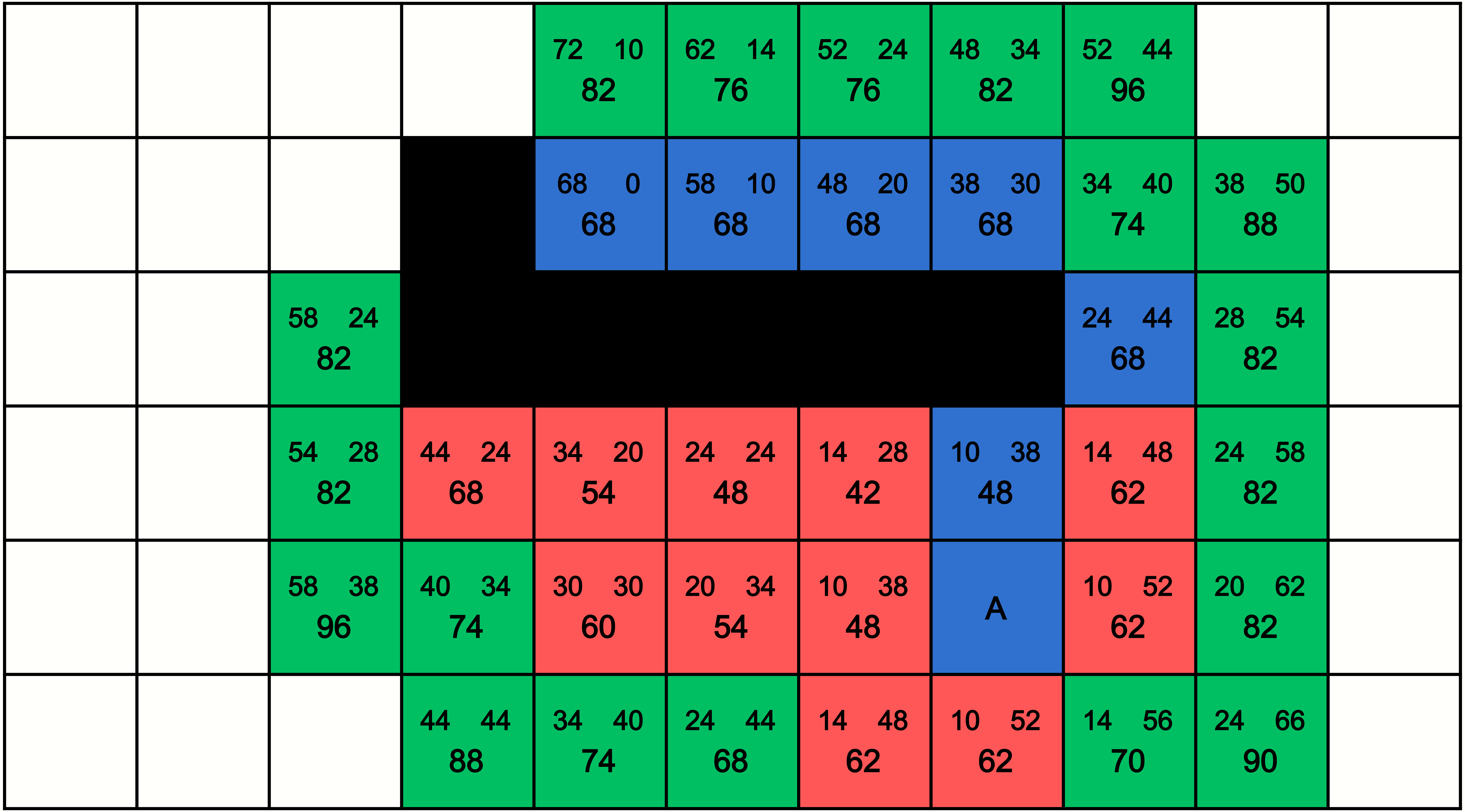

B
38 30 68
34 40 74
38 50 88
58 24 82
24 44 68
28 54 82
54 28 82
44 24 68
34 20 54
24 24 48
14 28 42
10 38 48
14 48 62
24 58 82
58 38 96
40 34 74
30 30 60
20 34 54
10 38 48
A
10 52 62
20 62 82
44 44 88
34 40 74
24 44 68
14 48 62
10 52 62
14 56 70
24 66 90


52 24 76
48 34 82
52 44 96
B
48 20 68
38 30 68
34 40 74
38 50 88
58 24 82
24 44 68
28 54 82
54 28 82
44 24 68
34 20 54
24 24 48
14 28 42
10 38 48
14 48 62
24 58 82
58 38 96
40 34 74
30 30 60
20 34 54
10 38 48
A
10 52 62
20 62 82
44 44 88
34 40 74
24 44 68
14 48 62
10 52 62
14 56 70
24 66 90


62 14 76
52 24 76
48 34 82
52 44 96
B
58 10 68
48 20 68
38 30 68
34 40 74
38 50 88
58 24 82
24 44 68
28 54 82
54 28 82
44 24 68
34 20 54
24 24 48
14 28 42
10 38 48
14 48 62
24 58 82
58 38 96
40 34 74
30 30 60
20 34 54
10 38 48
A
10 52 62
20 62 82
44 44 88
34 40 74
24 44 68
14 48 62
10 52 62
14 56 70
24 66 90



72 10 82
62 14 76
52 24 76
48 34 82
52 44 96
68 0 68
58 10 68
48 20 68
38 30 68
34 40 74
38 50 88
58 24 82
24 44 68
28 54 82
54 28 82
44 24 68
34 20 54
24 24 48
14 28 42
10 38 48
14 48 62
24 58 82
58 38 96
40 34 74
30 30 60
20 34 54
10 38 48
A
10 52 62
20 62 82
44 44 88
34 40 74
24 44 68
14 48 62
10 52 62
14 56 70
24 66 90


72 10 82
62 14 76
52 24 76
48 34 82
52 44 96
68 0 68
58 10 68
48 20 68
38 30 68
34 40 74
38 50 88
58 24 82
24 44 68
28 54 82
54 28 82
44 24 68
34 20 54
24 24 48
14 28 42
10 38 48
14 48 62
24 58 82
58 38 96
40 34 74
30 30 60
20 34 54
10 38 48
A
10 52 62
20 62 82
44 44 88
34 40 74
24 44 68
14 48 62
10 52 62
14 56 70
24 66 90