Week 4

Display replies in nested form

Settings

The cut-off date for posting to this forum is reached so you can no longer post to it.

Week 4

by Romana Riyaz (Instructor) - Thursday, 20 June 2024, 10:29 AM

In Chapter 11 of the Shaffer text, we are introduced to Kruskal's Algorithm for determining a Minimum Spanning Tree within a graph. This algorithm is described as a greedy algorithm. Describe in your own words how Kruskal's algorithm works and be sure to articulate in your description what makes Kruskal's algorithm 'greedy'.

Include one or two examples to explain your thought process to show what is occurring and how the methodology works. Use APA citations and references for any sources used.

81 words

Permalink

Re: Week 4

by Cherkaoui Yassine - Friday, 12 July 2024, 3:03 PM

Kruskal's Algorithm is a process used to find the Minimum Spanning Tree (MST) of a graph. A Minimum Spanning Tree is a subset of the edges that connects all vertices in the graph with the smallest possible total edge weight and no cycles.

How Kruskal's Algorithm Works

1. Sort the Edges by Weight: Start by sorting all the edges in the graph in ascending order based on their weights.

2. Initialize the MST: Begin with an empty MST.

3. Add the Smallest Edge: Add the smallest edge to the MST if it doesn't form a cycle with the edges already in the MST. This can be checked using the Union-Find data structure.

4. Repeat Until MST is Complete: Continue adding the next smallest edge that does not form a cycle, until the MST contains exactly $V - 1$ edges, where V is the number of vertices in the graph.

Why Kruskal's Algorithm is 'Greedy': The algorithm is termed 'greedy' because at each step, it picks the smallest available edge that connects two vertices without forming a cycle. This local optimization strategy is characteristic of greedy algorithms, which aim to find the global optimum by making a series of locally optimal choices.

Example 1: Simple Graph Consider a graph with four vertices (A, B, C, D) and the following edges with their weights:

(A - B): 1 / (A - C): 3 / (B - C): 2 / (B - D): 4 / (C - D): 5

Steps: 1. Sort the edges: (A - B): 1 / (B - C): 2 / (A - C): 3 / (B - D): 4 / (C - D): 5

2. Add edges: Start with (A - B) (1) / Add (B - C) (2) / Skip (A - C) (3) as it forms a cycle with (A - B) and (B - C) / Add (B - D) (4)

The MST includes the edges (A - B), (B - C), and (B - D), with a total weight of 7.

Example 2: More Complex Graph Consider a graph with vertices and these edges: (A - B): 1 / (A - C): 4 / (B - C): 3 / ([illegible]: 2 / (C - D): 5 / (C - E): 6 / (D - E): 7

Steps:

1. Sort the edges: (A - B): 1 / (B - D): 2 / (B - C): 3 / (A - C): 4 / (C - D): 5 / (C - E): 6 / (D - E): 7

2. Add edges:

- Start with (A - B) (1)

- Add (B - D) (2)

- Add (B - C) (3)

- Skip (A - C) (4) and (C - D) (5) as they form cycles

- Add (C - E) (6)

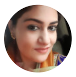
The MST includes the edges (A - B), (B - D), (B - C), and (C - E), with a total weight of 12.

References

Shaffer, C. A. (2012). Data Structures and Algorithm Analysis. Dover Publications.

504 words

Permalink Show parent



Re: Week 4
by Romana Riyaz (Instructor) - Friday, 12 July 2024, 11:36 PM

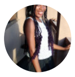
Cherkaoui,
Thank you for your submission. Your explanation of Kruskal's Algorithm is clear and concise, effectively outlining the steps involved in finding the Minimum Spanning Tree (MST). The description of why Kruskal's Algorithm is considered greedy adds valuable context, highlighting its local optimization strategy. The inclusion of two examples, one simple and one more complex, helps in illustrating the algorithm's application and its decision-making process. Providing references strengthens the credibility of your explanation. To enhance the feedback, consider mentioning potential drawbacks or limitations of Kruskal's Algorithm, such as its efficiency with respect to different types of graphs.
Best,
Romana
99 words

Permalink Show parent

Re: Week 4
by Naqaa Alawadhi - Sunday, 14 July 2024, 4:52 PM

Good job
2 words

Permalink Show parent



Re: Week 4
by Akomolafe Ifedayo - Monday, 15 July 2024, 5:07 PM

Hi Cherkaoui, great work. You described Kruskal's algorithm and explained why it is termed greedy. You also provided some examples and steps involved in finding the Minimum Spanning Tree (MST). Keep it up.
33 words

Permalink Show parent

Re: Week 4

by Benjamin Chang - Tuesday, 16 July 2024, 1:46 AM

Hi, Cherkaoui.
Thank you for sharing our initial discussion this week; you are the first poster, and your responses are completely correct, easy to comprehend, and I enjoy them. Keep it up!
Benjamin
33 words

Permalink Show parent

Re: Week 4
by Moustafa Hazeen - Tuesday, 16 July 2024, 3:20 AM

Your explanation of Kruskal’s Algorithm is clear and covers the basic steps well. The examples illustrate the process, but there are a few inaccuracies and missing details. Adding a bit more about the Union-Find data structure and checking your example calculations could strengthen your explanation. Also, make sure your APA citation follows the correct format. Good job overall!
58 words

Permalink Show parent

Re: Week 4
by Fadi Al Rifai - Tuesday, 16 July 2024, 4:54 PM

Hi Cherkaoui,
Thank you for your thoughtful contribution to a detailed explanation of how Kruskal’s algorithm works, and I like your description of examples.
Keep it up.
27 words

Permalink Show parent

Re: Week 4
by Siraajuddeen Adeitan Abdulfattah - Tuesday, 16 July 2024, 7:33 PM

Hi Cherkaoui,

You have provided a well articulated response on Kruskal' Algorithm, the steps it uses in determining the MST and explanation on what makes Kruskal's Algorithm greedy. You gave multiple examples to show Kruskal's Algorithm application and how it functions. Overall, you presented your response in a simple yet, educative and informative manner.
54 words

Permalink Show parent

Re: Week 4
by Nour Jamaluddin - Thursday, 18 July 2024, 5:33 AM

Perfect job.
You have done a complete search. Your post met all the requirements. I learned from your explanation.
Thanks for sharing.
22 words

Permalink Show parent

Re: Week 4
by Chong-Wei Chiu - Thursday, 18 July 2024, 8:59 AM

Hello, Cherkaoui Yassine. Thank you for sharing your perspectives on Kruskal’s Algorithm. You clearly explain how Kruskal’s Algorithm works and why it is considered a type of greedy algorithm. Finally, you illustrate these concepts with a simple, step-by-step example. Overall, your post meets all the requirements of this discussion.

49 words

Permalink Show parent

Re: Week 4

by Moustafa Hazeen - Saturday, 13 July 2024, 1:34 AM

Kruskal's algorithm employs a greedy approach to identify the minimum-cost spanning tree. In this method, the graph is viewed as a collection of trees initially, where each node represents a separate tree. To connect these trees into a spanning tree, the algorithm selects edges based on their weight in ascending order. Each chosen edge is added to the spanning tree if it does not create a cycle with the existing tree structure, ensuring it adheres to the properties of a minimum spanning tree (MST). This process involves sorting edges by weight, selecting the smallest edge, verifying it doesn't form a cycle, and repeating until the spanning tree contains V−1V-1V−1 edges (Tutorial points, n.d.).

Top of Form

Bottom of Form

(geeksforgeeks, 2021).

There are 9 vertices and 14 edges in this graph. As a result, the smallest spanning tree will have

(9 – 1) = 8 edges.

We will have this after sorting.

After sorting:

Weight	Src	Dest
1	7	6
2	8	2
2	6	5
4	0	1
4	2	5
6	8	6
7	2	3
8	0	7
8	1	2
9	3	4

10	5	4
11	1	7
14	3	5

Edge 7 – 6 is chosen without forming a cycle, so it is added to the set. Similarly, edges 8 – 2, 6 – 5, 0 – 1, and 2 – 5 are also included because they do not create cycles. However, selecting edge 8 – 6 would create a cycle, so it is rejected. Edge 2 – 3 is added as it does not form a cycle. Edge 7 – 8 is omitted because it would create a cycle. Edge 0 – 7 is added to the set for the same reason as before. Edge 1 – 2 is not included. Finally, edge 3 – 4 is added because it does not form a cycle (Tutorial points, n.d.).

(geeksforgeeks, 2021).

The algorithm stops at this point because the number of included edges reaches V−1V - 1V−1. Why is it considered greedy? A greedy algorithm is characterized by selecting the best possible solution or choice at each step of a problem (Shaffer, 2010). Kruskal's Algorithm exemplifies this by choosing edges with the lowest weight, ensuring it finds the most optimal path, which aligns perfectly with the concept of being greedy.

References

Geeksforgeeks. (2021). Kruskal's Minimum Spanning Tree Algorithm | Greedy Algo-2. https://www.geeksforgeeks.org/kruskals-minimum-spanning-tree-algorithm-greedy-algo-2/

Shaffer, C. A. (2010). A Practical Introduction to Data Structures and Algorithm Analysis, Third Edition (C++ Edition) – Chapter 11: Graphs. http://www.cse.ust.hk/~dekai/271/notes/L07/L07.pdf

Tutorial points. (n.d.). Kruskal's Spanning Tree Algorithm. https://www.tutorialspoint.com/data_structures_algorithms/kruskals_spanning_tree_algorithm.htm

415 words

Re: Week 4
by Romana Riyaz (Instructor) - Sunday, 14 July 2024, 12:49 AM

Moustafa,
Your explanation of Kruskal's algorithm is clear and well-structured, effectively describing the process of finding a minimum-cost spanning tree using a greedy approach. The step-by-step explanation of how edges are selected and added to the spanning tree is very helpful. The inclusion of a specific example with vertices and edges, along with weights, enhances understanding. Ensure consistent formatting throughout the document. For instance, maintain uniformity in the use of bold or italics for key terms. There are some redundant labels like "Top of Form" and "Bottom of Form" that can be removed. You effectively describe why Kruskal's algorithm is considered greedy by emphasizing the selection of the lowest

weight edges at each step. References are correctly cited, but ensure all sources are formatted consistently in your references list. Provide more context for each reference to enhance credibility and make it easier for readers to follow up on the sources. There are minor grammatical issues, such as unnecessary repetition of "V−1V-1V−1" and incomplete sentences. Proofread for such errors to ensure readability. The example is detailed and demonstrates the application of Kruskal's algorithm well. However, consider adding a final summary of the chosen edges to make it more comprehensive.
Best,
Romana
200 words

Permalink Show parent

Re: Week 4
by Cherkaoui Yassine - Monday, 15 July 2024, 2:59 PM

Moustafa, I wanted to drop you a quick note to say excellent job on your discussion post! Your answer is not only clear but also very well-written. It's evident that you put thought and effort into it, and it really shines through. Keep up the great work!
47 words

Permalink Show parent

Re: Week 4
by Benjamin Chang - Tuesday, 16 July 2024, 1:49 AM

Hi, Moustafa.
This is a really good explanation and highly professional examination of Kruskal's algorithm step by step. I like how you educate us how to become acquainted with this algorithm; good work!
Benjamin
34 words

Permalink Show parent

Re: Week 4
by Siraajuddeen Adeitan Abdulfattah - Tuesday, 16 July 2024, 7:43 PM

Hi Moustafa,
You provided a concise and detailed explanation on Kruskal's Algorithm and also described what makes it greedy. Your example shows Kruskal's Algorithm application and how it operates, offering step-by-step explanation of how MST is built. Inclusion of in-text citations and references gives strong support to your explanation
49 words

Permalink Show parent

Re: Week 4
by Mejbaul Mubin - Wednesday, 17 July 2024, 4:43 AM

Hi Moustafa,

Your explanation of Kruskal's algorithm is thorough and well-structured, offering a clear walkthrough of how it constructs a minimum-cost spanning tree using a greedy approach. The inclusion of a detailed example with vertices, edges, and weights enhances comprehension. Ensure consistent formatting throughout and address minor grammatical issues for improved readability. Adding a final summary of selected edges would further enhance the clarity of your explanation.
67 words

Permalink Show parent

Re: Week 4
by Nour Jamaluddin - Thursday, 18 July 2024, 5:36 AM

Well done.
You understand the concept of Kruskal's algorithm. I liked adding the photos to illustrate your ideas. Your writing is clear and simple.
Thanks a lot!
27 words

Permalink Show parent

Re: Week 4

by Chong-Wei Chiu - Thursday, 18 July 2024, 9:05 AM

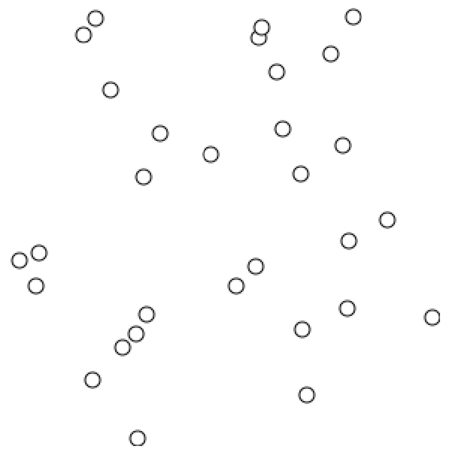
Hello, Moustafa Hazeen. Thank you for sharing your opinion on this topic. You clearly explain the concepts of Kruskal's Algorithm, and you use a simple graph to illustrate your example, making it easy to understand. I think that's amazing!
39 words

Permalink Show parent

Re: Week 4

by Anthony Jones - Saturday, 13 July 2024, 7:34 AM

Kruskal's algorithm determines the minimum spanning tree by repeatedly adding the next shortest/least expensive edge in the entire graph that does not create a cycle with the structure it has already constructed until all the vertices have been added. This works since we know that we will be using the least cost edges to connect the vertices and since we know we cannot have cycles in our minimum spanning tree (otherwise it would not be a tree). The algorithm usually starts out creating multiple disconnected sub-graphs but gradually connects them until they form a unified tree.
Here is a visualization of Kruskal's algorithm:



(Ji, 2016) CC 4.0 License

Kruskal's algorithm is usually implemented with a priority queue or some form of list that is sorted by edge cost that it loops through. This allows it to evaluate the next shortest edge efficiently. However, when evaluating an edge, it can be tricky to determine whether an edge creates a cycle or not since there may be several different sets in progress. Fortunately, *Lecture 8: Kruskal's MST Algorithm* (n.d.) gives us a nice simple way of identifying whether an edge creates a cycle. If two edges are in the same set (i.e. we have connected them by edges), it makes a cycle. The reason this works is rather simple: if we have already mapped a route between the vertices (ex, *a* and *f*), then we can use our already existing route (ex. $a \rightarrow b \rightarrow c \rightarrow e \rightarrow f$) to get from a to *f*, then we can go from *f* to *a* using the edge we are considering and now we have a cycle $a \rightarrow b \rightarrow c \rightarrow e \rightarrow f \rightarrow a$. This can be illustrated below:

Here is some pseudo-code for the algorithm:

```
g = graph
mst = new Tree()
edgeOptions = new PriorityQueue(g.edges)
while (len(mst.Vertices) < len(g.Vertices)):
    e = edgeOptions.poll() //get the next smallest edge while simultaneously removing it
    if e does not create cycle in mst:
        mst.add(e)
```

So, what makes Kruskal's algorithm greedy? Krusakl's algorithm is greedy because it constantly picks the evaluates the best option of the existing edges. This follows the greedy principle of choosing the best option locally and allowing that to contribute to the best global option. We know that the Minimum spanning tree is going to be comprised of the least cost edges to connect vertices. So, by greedily selecting the least cost edge that connects two unconnected vertices, we know we will create the MST step-by-step. That's what makes Kruskal's algorithm greedy: it constructs the minimum spanning tree (the global best) step-by-step by choosing the least cost non-redundant edge (local best).

References

Ji, S. (2016, December 24). *A demo for Kruskal's algorithm to find minimum spanning tree on a 2D plane.* Wikimedia Commons.

Lecture 8: Kruskal's MST algorithm. (n.d.). Retrieved July 13, 2024, from https://www.cse.ust.hk/~dekai/271/notes/L08/L08.pdf

479 words

Permalink Show parent

Re: Week 4
by Romana Riyaz (Instructor) - Sunday, 14 July 2024, 12:52 AM

Anthony,
Thank you for your submission. Your explanation of Kruskal's algorithm is thorough and clearly articulated. Your introduction succinctly captures the essence of Kruskal's algorithm and sets the stage for the detailed explanation that follows.The step-by-step breakdown of the algorithm, including the visualization and pseudo-code, enhances understanding. Mentioning the visualization (File.gif) helps in illustrating the algorithm. However, ensure that the visualization is accessible or provide an alternative description if the file cannot be viewed directly. The explanation of why the algorithm is greedy is clear and effectively ties back to the core principles of greedy algorithms. The section on cycle detection is well-explained and provides a concrete example, making the concept easier to grasp.There are minor formatting issues, such as the misspelling of "Kruskal's" as "Krusakl's" in one instance. Consistent use of formatting for terms and emphasis will improve readability. For example, consistently italicizing or bolding key terms and algorithm names.The pseudo-code is a helpful addition, providing a clear, concise summary of the algorithm's implementation. Ensure the pseudo-code is formatted correctly (e.g., consistent indentation) to enhance readability. References are correctly cited, but ensure that they are consistently formatted. Consider providing more context for each reference to enhance credibility and make it easier for readers to follow up on the sources.
Best,
Romana
212 words

Permalink Show parent

Re: Week 4
by Naqaa Alawadhi - Sunday, 14 July 2024, 4:53 PM

Good j
2 words

Permalink Show parent

Re: Week 4
by Cherkaoui Yassine - Monday, 15 July 2024, 2:59 PM

Hey, just wanted to give you props for your discussion post—it's fantastic! Your response is clear, articulate, and well-structured. It's evident you know your stuff and put in the effort to communicate effectively. Keep up the great work!
39 words

Permalink Show parent

Re: Week 4

by Akomolafe Ifedayo - Monday, 15 July 2024, 5:11 PM

Hi Anthony, I enjoyed going through your submission. Your use of visualization and images to describe the Kruskal's algorithm was interesting. You also provided some pseudo-code and explained why the algorithm is termed greedy.

34 words

Permalink Show parent

Re: Week 4

by Benjamin Chang - Tuesday, 16 July 2024, 1:51 AM

Hi, Anthony.
I appreciate your discussion post because you use an image that illustrates Kruskal's algorithm very well. Everyone uses an image to teach people about Kruskal's algorithm, which is extremely interesting because everyone uses a different image, yet all responses are right! Keep it up!
Benjamin

47 words

Permalink Show parent

Re: Week 4

by Siraajuddeen Adeitan Abdulfattah - Tuesday, 16 July 2024, 7:55 PM

Hi Anthony,

You provided a clear explanation of Kruskal's Algorithm, explaining how its operates and the steps involved in creating MST. The visualization of Kruskal's Algorithm provided is nice but will benefit readers even more if there is an explanation for the visualization. You also correctly stated what makes Kruskal's Algorithm greedy and you offer a valid explanation on how to identify if an edge forms a cycle, connecting back to the vertex where it started.

76 words

Permalink Show parent

Re: Week 4

by Nour Jamaluddin - Thursday, 18 July 2024, 5:38 AM

Very good job.
Your answer is correct and met the required. The citations you used were helpful and appropriate. I liked your post organization.
Good luck

26 words

Permalink Show parent

Re: Week 4
by Jerome Bennett - Thursday, 18 July 2024, 7:41 AM

Greetings Anthony,

You did a great job explaining Kruskal's Algorithm and how it works to create a Minimum Spanning Tree (MST). The visualization is helpful, but it would be even better with a bit of explanation to go along with it. You are also spot-on as to why Kruskal's Algorithm is considered greedy and clearly explained how to check if an edge forms a cycle by connecting back to its starting vertex.

72 words

Permalink Show parent

Re: Week 4
by Benjamin Chang - Saturday, 13 July 2024, 11:14 PM

Kruskal's algorithm finds the minimum spanning tree by selecting the minimum weighted edges in increasing order of weight, ensuring no cycles are formed until the tree is complete. According to GeeksforGeeks (2024), it starts with all vertices as separate trees and then repeatedly chooses the minimum weight edge. The best, average, and worst-case time complexity is O(E log E) and the space complexity is O(V+E).

The reason we call this a greedy algorithm is that it only considers choosing the shortest edge with the lowest weight. The greedy aspect lies in its approach to not creating cycles while connecting trees into a single tree. This method is efficient in optimizing all paths. Let's break down how Kruskal’s algorithm works:

List and sort edges: First, we need to list all edges and sort them in ascending order. For example, sort the edges from weight 1, which is the minimum weight, to weight 2, weight 3, and so on.

Build the MST: Start with the smallest edge (weight 1) and add it to the MST, ensuring that it does not form a cycle. If adding an edge creates a cycle, discard that edge.

Continue the process: Repeat this process until the MST contains V−1 edges, ensuring that all vertices are connected in the MST.

Finally, I am curious about how the time complexity of Kruskal's algorithm works. Here is my personal opinion. The time complexity is O(E log E), where E is the number of edges. The sorting time of the edges is O(E log E), which accounts for sorting all the edges by weight.

After sorting, we start with a loop to process each edge. In each iteration, we select one edge, and the total number of iterations will not exceed the number of edges. Therefore, the overall complexity consists of two main components:

Checking whether the two vertices of an edge belong to the same tree, which is typically O(log V) per operation using a union-find data structure.

Merging two trees, which is also O(log V) per operation.

Thus, the overall time complexity is dominated by the sorting step, O(E log E), plus the union-find operations over all edges, which is O(E log V). However, since V≤ E O(log V) is at most O(logE). So, the total time complexity is O(E log E).

Let me illustrate how Kruskal’s algorithm works using an example from a Wikimedia animation. Consider we have the following edges: ae, cd, ab, be, bc, ec, and ed. We list and sort their weights from 1 to 7.

Start with the smallest edge: Begin with the edge ae with weight 1. Since it does not form a cycle, we include it in the MST.

Continue to the next edge: Move to edge cd with weight 2. It also does not form a cycle, so we include it in the MST.

Proceed to the next edge: Next is edge ab with weight 3. It does not form a cycle, so we include it in the MST.

Check the next edge: Now consider edge be with weight 4. This edge would form a cycle if included, so we discard it.

Move to the next edge: Next is edge bc with weight 5. Including this edge does not form a cycle, so we add it to the MST.

We stop here because we have included V−1 edges (where V is the number of vertices). Therefore, we have successfully found the minimum spanning tree for this graph.

Reference

T kruskal en.gif - Wikimedia Commons*. (2014, January 30).

orGeeks. (2024, February 19). *Time and space complexity analysis of Kruskal Algorithm*. GeeksforGeeks. https://www.geeksforgeeks.org/time-and-space-complexity-analysis-of-kruskal-algorithm/

, C. A. (2010). *A Practical Introduction to data Structures and algorithm Analysis third edition (C++ Version)* (3rd ed.). https://people.cs.vt.edu/~shaffer/Book/C++3e20100119.pdf

621 words

Permalink Show parent

Re: Week 4

by Naqaa Alawadhi - Sunday, 14 July 2024, 4:55 PM

excellent

1 words

Permalink Show parent

Re: Week 4

by Romana Riyaz (Instructor) - Sunday, 14 July 2024, 11:13 PM

Benjamin,
Thank you for your submission. The explanation is clear and well-structured, breaking down the steps of Kruskal's algorithm in a logical manner. It covers the essential aspects of the algorithm, including the greedy approach, sorting of edges, and the union-find data structure. The discussion on time complexity is thorough and explains why the overall complexity is dominated by the sorting step. Using an example to illustrate the algorithm's steps is helpful for understanding the practical application.
Best,
Romana

79 words

Permalink Show parent

Re: Week 4

by Cherkaoui Yassine - Monday, 15 July 2024, 2:59 PM

Benjamin, I wanted to take a moment to acknowledge your discussion post—it's really well-done! Your answer is clear, concise, and well-articulated. It's evident you put thought and care into your response, and it's paying off. Keep up the excellent work!

41 words

Permalink Show parent

Re: Week 4
by Akomolafe Ifedayo - Monday, 15 July 2024, 5:13 PM

Hi Benjamin, great work. Your use of an image to visually explain the concept of the Kruskal's algorithm was engaging to see. You also broke down how the algorithm works, and why it is termed greedy. Keep it up.

39 words

Permalink Show parent

Re: Week 4
by Moustafa Hazeen - Tuesday, 16 July 2024, 3:24 AM

Your explanation of Kruskal’s Algorithm is mostly accurate and well-organized. You cover the steps and complexity analysis effectively. However, the example could be clearer, and there are minor citation errors. Ensure the example edges and weights are well-defined, and double-check APA formatting for citations. Good work overall!

47 words

Permalink Show parent

Re: Week 4
by Siraajuddeen Adeitan Abdulfattah - Tuesday, 16 July 2024, 8:04 PM

Hi Benjamin,

You provided a thorough explanation on Kruskal's Algorithm and also stated why it is considered greedy. Your step-by-step explanation of the processes involved in Kruskal's Algorithm while creating the MST makes it easy to understand and your addition of animation shows the picture of what goes in the steps explained earlier in your post. You gave a clear and good account of what Kruskal's Algorithm is about and it application.

72 words

Permalink Show parent

Re: Week 4
by Anthony Jones - Wednesday, 17 July 2024, 2:14 AM

Hello,

Good post! I liked the animation you used to illustrate Kruskal's algorithm and your step-by-step explanation. Your explanation of the "greediness" of the algorithm, wasn't entirely clear. I was wondering if you could expand upon it. How does it fit with the greedy approach: optimize locally to optimize globally?

Also, how do we determine if an edge creates a cycle? How expensive is it to check and how does this effect the algorithm's time complexity?

God bless!

Anthony

79 words

Permalink Show parent

Re: Week 4
by Jerome Bennett - Thursday, 18 July 2024, 7:37 AM

Greetings Benjamin,

Your answer got straight to the point but still covered everything important. I can tell you put a lot of effort into your presentation. The Wikimedia animation surely drove home my understanding of Kruskal’s algorithm. Nice work!
39 words

Permalink Show parent

Re: Week 4
by Naqaa Alawadhi - Sunday, 14 July 2024, 4:12 PM

Kruskal's Algorithm is a greedy algorithm used to find the Minimum Spanning Tree (MST) of a graph. It works by selecting edges in ascending order of their weights while ensuring that no cycle is formed in the process. The algorithm starts with individual vertices as separate trees and then iteratively adds the shortest edge that does not form a cycle until all vertices are connected.

The "greedy" aspect of Kruskal's Algorithm lies in its strategy of always choosing the smallest edge available at each step without reconsidering previous choices. This approach aims to minimize the total weight of the spanning tree without backtracking or revising decisions made earlier.

For example, consider a graph with vertices A, B, C, D and edges with weights: AB=2, AC=3, AD=4, BC=5, BD=1, CD=6. Kruskal's Algorithm would start by selecting edge BD (weight 1), then AB (weight 2), followed by AC (weight 3) to form the MST with total weight 6.

Shaffer, C.A. (Year). Title of Book. Publisher.
163 words

Permalink Show parent

Re: Week 4
by Michael Oyewole - Monday, 15 July 2024, 4:21 AM

Hi Naqaa,
Thank you for the post. You stated that a greedy approach called Kruskal's approach is used to determine a graph's Minimum Spanning Tree (MST). And that it chooses edges according to weight in ascending order, making sure that no cycle forms in the process. And you stated that until all vertices are connected, the algorithm begins with each vertex as a separate tree and repeatedly adds the shortest edge that does not generate a cycle. Thanks so much!
80 words

Permalink Show parent

Re: Week 4
by Romana Riyaz (Instructor) - Wednesday, 17 July 2024, 12:28 AM

Naqaa,
Thank you for your response. The description of Kruskal's Algorithm effectively captures its essence as a greedy algorithm used to find the Minimum Spanning Tree (MST) of a graph. The explanation of how the algorithm works, by selecting edges in ascending order of their weights while avoiding cycles, is clear and concise. The example provided with vertices A, B, C, D and their corresponding edges helps illustrate the process and reinforces understanding. Additionally, highlighting the "greedy" aspect of the algorithm, where the smallest edge is chosen at each step without reconsidering previous choices, effectively conveys its strategy. Overall, this explanation is well-structured and informative, making the concept of Kruskal's Algorithm accessible and easy to grasp but you could have provided more details and the reference added is also not done correctly.

Best,
Romana
134 words

Permalink Show parent

Re: Week 4
by Fadi Al Rifai - Sunday, 14 July 2024, 9:59 PM

Kruskal's Algorithm is a classic method for finding the Minimum Spanning Tree (MST) of a graph. The MST of a graph is a subset of the edges that connects all the vertices together without any cycles and with the minimum possible total edge weight.

According to Geeks for Geeks, "In Kruskal's algorithm, sort all edges of the given graph in increasing order. Then it keeps on adding new edges and nodes in the MST if the newly added edge does not form a cycle. It picks the minimum weighted edge at first and the maximum weighted edge at last. Thus, we can say that it makes a locally optimal choice in each step in order to find the optimal solution. Hence this is a Greedy Algorithm." (Geeks for Geeks, 2023).

How Kruskal's Algorithm Works:

1. **Sort Edges**: Start by sorting all the edges in the graph by their weight in ascending order.
2. **Initialize Subsets**: Create a separate subset for each vertex. This helps in detecting cycles in the growing spanning tree.
3. **Add Edges**: Pick the smallest edge and check if it forms a cycle with the spanning tree formed so far. If the edge doesn't form a cycle, include it in the MST.
4. **Repeat**: Continue adding edges until there are exactly V−1 edges where V is the number of vertices in the graph in the MST.

Why Kruskal's Algorithm is Greedy:

Kruskal's Algorithm is considered greedy because, at each step, it chooses the smallest available edge without considering the global structure of the graph. The algorithm makes a local optimal choice (the smallest edge) with the hope that these local optima will lead to a globally optimal solution (the MST).

Example 1:

I'll consider a graph with 4 vertices (A, B, C, D) and the following edges with weights:

- A-B (1)
- A-C (3)
- B-C (3)
- B-D (6)
- C-D (4)

Steps:

1. **Sort edges**: A-B (1), A-C (3), B-C (3), C-D (4), B-D (6)
2. **Initialize subsets**: {A}, {B}, {C}, {D}
3. **Add A-B**: No cycle, add to MST.

- MST: {A-B}

Add A-C: No cycle, add to MST.
- MST: {A-B, A-C}

Add B-C: Forms a cycle (A-B-C-A), discard. **Add C-D**: No cycle, add to MST.
- MST: {A-B, A-C, C-D}

Final MST: {A-B, A-C, C-D} with total weight 1 + 3 + 4 = 8.

Example 2:

I'll consider a graph with 5 vertices (A, B, C, D, E) and the following edges with weights:

- A-B (2)
- A-D (1)
- B-C (3)
- B-D (2)
- C-E (4)
- D-E (5)
- D-C (3)

Steps:

1. **Sort edges**: A-D (1), A-B (2), B-D (2), B-C (3), D-C (3), C-E (4), D-E (5)
2. **Initialize subsets**: {A}, {B}, {C}, {D}, {E}
3. **Add A-D**: No cycle, add to MST.

- MST: {A-D}

Add A-B: No cycle, add to MST.
- MST: {A-D, A-B}

Add B-D: Forms a cycle (A-D-B-A), discard. **Add B-C**: No cycle, add to MST.
- MST: {A-D, A-B, B-C}

Add D-C: Forms a cycle (A-D-C-B-A), discard. **Add C-E**: No cycle, add to MST.
- MST: {A-D, A-B, B-C, C-E}

Final MST: {A-D, A-B, B-C, C-E} with total weight 1 + 2 + 3 + 4 = 10.

In my opinion, Kruskal's Algorithm efficiently finds the MST by continually making the locally optimal (smallest weight edge) choice. The algorithm's greedy nature ensures that the overall tree has the minimum total weight while connecting all vertices without forming cycles.

References:

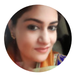
Geeks for Geeks. (2023, July 31). Kruskal's minimum spanning tree (MST) algorithm. https://www.geeksforgeeks.org/kruskals-minimum-spanning-tree-algorithm-greedy-algo-2/

Chapter 11 Graphs, Sections 11.1 – 11.3 in A Practical Introduction to Data Structures and Algorithm Analysis by Clifford A. Shaffer.
Read Lecture 8: Lecture 8: Kruskal's MST Algorithm available at http://www.cse.ust.hk/~dekai/271/notes/L08/L08.pdf

Read the Wikipedia article on Kruskal's algorithm paying special attention to the pictoral representation of how Kruskal's algorithm creates a minimum spanning tree from a collection of graphs that are assembled together. Available at http://en.wikipedia.org/wiki/Kruskal's_algorithm

631 words

Permalink Show parent



Re: Week 4
by Romana Riyaz (Instructor) - Sunday, 14 July 2024, 11:20 PM

Fadi,
Thank you for your submission. The explanation covers the key aspects of Kruskal's Algorithm, including its purpose, steps, and why it is considered a greedy algorithm. The step-by-step breakdown of how Kruskal's Algorithm works is clear and easy to follow. This helps readers understand the process from start to finish. The inclusion of two detailed examples

provides practical illustrations of the algorithm in action, which enhances comprehension. Adding a personal opinion on the efficiency of the algorithm offers a reflective perspective, making the explanation more engaging. Citing multiple sources, including Geeks for Geeks and academic materials, adds credibility and provides readers with resources for further study However, you should follow proper APA guidelines.
Best,
Romana
116 words

Permalink Show parent

Re: Week 4
by Michael Oyewole - Monday, 15 July 2024, 4:15 AM

Hi Fadi,
Thank you for your post. You stated that the sorting edge in the provided graph according to Kruskal's algorithm is in ascending order. You also stated that if the newly added edge does not form a cycle, it then continues to add additional edges and nodes to the MST. And it selects the highest weighted edge last and the lowest weighted edge first. Thanks for sharing.
68 words

Permalink Show parent

Re: Week 4
by Anthony Jones - Wednesday, 17 July 2024, 2:26 AM

Hello,

Good post! I liked your explanation for the greediness of the algorithm -- it is very concise and accurate. I also liked your step-by-step explanation of Kruskal's algorithm. How do we check if an edge creates a cycle and how do we know it shouldn't be in the minimum spanning tree?

God bless!

Anthony
55 words

Permalink Show parent

Re: Week 4
by Mahmud Hossain Sushmoy - Monday, 15 July 2024, 12:22 AM

Kruskal's algorithm is a greedy algorithm used to find the Minimum Spanning Tree (MST) of a graph. The MST is a subset of the edges that connects all vertices in the graph without any cycles and with the minimum possible total edge weight.

Kruskal's algorithm works as follows: First, sort all the edges in the graph by their weight in non-decreasing order. Then, create a subset for each vertex, initially treating each vertex as its own subset. Starting with the smallest edge, include it in the MST as long as it doesn't create a cycle with the edges already added. Continue adding the next smallest edge that does not form a cycle until the MST contains n-1 edges where n is the number of vertices).

This algorithm is considered greedy because at each step, it chooses the smallest available edge without considering the overall structure of the MST. By making a series of locally optimal choices (selecting the smallest edge), the algorithm aims to achieve a globally optimal solution (Shaffer, 2011).

Example

Consider a graph with vertices A, B, C, D, E, and F and edges with weights as follows: (A, B) = 1, (B, C) = 2, (C, D) = 1, (D, E) = 3, (E, F) = 1, and (C, F) = 2.

First, sort the edges: (A, B), (C, D), (E, F) with weight 1; (B, C), (C, F) with weight 2; (D, E) with weight 3. Start with the smallest edge (A, B), which does not form a cycle, and add it to the MST. Then, add (C, D) and (E, F), both of which do not form cycles. Next, add (B, C), which also does not form a cycle. Edges (C, F) and (D, E) are skipped as they would form cycles.

The resulting MST includes the edges (A, B), (C, D), (E, F), and (B, C) with a total weight of 5. This process demonstrates how Kruskal's algorithm builds the MST by always choosing the smallest edge available, showcasing its greedy nature.

References

Shaffer, C. (2011). *A Practical Introduction to Data Structures and Algorithm Analysis*. Blacksburg: Virginia. Tech.

354 words

Permalink Show parent

Re: Week 4

by Michael Oyewole - Monday, 15 July 2024, 4:03 AM

Hi Mahmud,
Thank you for your post. I agreed with you that the algorithm mentioned is greedy because of the displayed steps. Also, brute force algorithms work by methodically examining each potential answer in order to choose the best one. Although this approach is simple, it may not be effective when dealing with big problem sizes. Thanks for the post.
60 words

Permalink Show parent

Re: Week 4

by Manahil Farrukh Siddiqui - Thursday, 18 July 2024, 3:49 AM

Hello Mahumd,

Your answer is well-structured. You have effectively illustrates how the algorithm works in practice.
16 words

Permalink Show parent

Re: Week 4

by Jerome Bennett - Thursday, 18 July 2024, 7:44 AM

Greetings Krustkal,
I like that your work is concise. I agree with you that the algorithm is greedy because of the steps it follows. Brute force algorithms, on the other hand, work by systematically checking each possible answer to find the best one. While this method is straightforward, it can be pretty inefficient for large problems.
56 words

Permalink Show parent

Re: Week 4

by Michael Oyewole - Monday, 15 July 2024, 3:10 AM

Kruskal's approach finds the minimum cost spanning tree by using the greedy technique. With this method, every node in the network represents a different tree, treating the graph as a forest. If a tree does not violate the MST criteria and has the lowest cost among all the options, it can only link to another tree. It involves the subsequent actions: The edges are sorted in a non-descending manner based on weight. Select the smallest possible edge. Examine whether it creates a cycle with the spanning tree that you have already made. If there isn't a cycle, include this edge. If not, discard it. Until the spanning tree has (V-1) edges, repeat the second test (Tutorial points, n.d.).

(Geeksforgeeks, 2021).

This graph has 14 edges and 9 vertices. The smallest spanning tree will therefore have (9 – 1) = 8 edges. This is what we'll have once we sort. Following sorting:

Weight	Src	Dest
1	7	6
2	8	2
2	6	5
4	0	1
4	2	5
6	8	6
7	2	3
7	7	8
8	0	7
8	1	2
9	3	4
10	5	4
11	1	7
14	3	5

Edges 7–6 do not constitute a cycle, thus we shall include them. Similarly, since no cycle was created, 8 – 2, 6 – 5, and 0 – 1, 2 – 5 will also be included. But if we choose 8–6, including this edge will cause a loop, therefore we'll throw it out. As no cycle is created, 2- 3 will be included. 7 and 8 will not be included because doing so will create a cycle. 0 through 7 will be included for the same previously stated purpose. 1 and 2 will be thrown away. Edges 3 and 4 will be included if they are selected since they do not create a cycle.

(geeksforgeeks, 2021).

This is where the algorithm stops because the total number of edges included is equal to (V – 1). Why is this acting so greedily? According to Shaffer (2010), a greedy algorithm is a strategic algorithm that finds the best possible answer or choice for a given situation. Because Kruskal's Algorithm chooses the edges with the least weight, which leads to the best optimal path, exactly what greedy means.

References

Geeksforgeeks. (2021). Kruskal's Minimum Spanning Tree Algorithm | Greedy Algo-2. https://www.geeksforgeeks.org/kruskals-minimum-spanning-tree-algorithm-greedy-algo-2/

Shaffer, C. A. (2010). A Practical Introduction to Data Structures and Algorithm Analysis, Third Edition (C++ Edition) – Chapter 11: Graphs. http://www.cse.ust.hk/~dekai/271/notes/L07/L07.pdf

Tutorial points. (n.d.). Kruskal's Spanning Tree Algorithm. https://www.tutorialspoint.com/data_structures_algorithms/kruskals_spanning_tree_algorithm.htm

417 words

Permalink Show parent

Re: Week 4
by Romana Riyaz (Instructor) - Tuesday, 16 July 2024, 12:37 AM

Michael,
Thank you for your response. The description of Kruskal's approach is clear and accurately captures the essence of the algorithm. The step-by-step process of sorting edges, selecting the smallest edge, checking for cycles, and including or discarding edges is well-explained. The example with 14 edges and 9 vertices effectively demonstrates the practical application of Kruskal's algorithm. The inclusion and exclusion of edges based on cycle formation are illustrated clearly.

Adding a visual representation of the graph before and after applying Kruskal's algorithm would enhance understanding. Diagrams showing the initial graph, the sorted edges, and the final minimum spanning tree (MST) can make the process more intuitive. While the text mentions checking for cycles, a brief explanation of how cycles are detected in Kruskal's algorithm (e.g., using the Union-Find data structure) would be beneficial for readers unfamiliar with the concept. The example jumps from edge inclusion/exclusion steps to conclusions without always explaining why certain edges are discarded (e.g., "1 and 2 will be thrown away"). Clarifying these decisions at each step can improve clarity.

Best,

Romana

176 words

Permalink Show parent

Re: Week 4

by Fadi Al Rifai - Tuesday, 16 July 2024, 4:55 PM

Hi Michael,

Good work, Thanks for sharing your explanation about how Kruskal's algorithm works, and I like your description of examples.

Keep it up.

24 words

Permalink Show parent

Re: Week 4

by Mejbaul Mubin - Monday, 15 July 2024, 5:37 AM

Kruskal's algorithm is a classic greedy algorithm used to find the Minimum Spanning Tree (MST) of a graph. A Minimum Spanning Tree is a subset of the edges in a connected, undirected graph that connects all the vertices together, without any cycles, and with the minimum possible total edge weight.

How Kruskal's Algorithm Works

Sort the Edges: First, all the edges of the graph are sorted in non-decreasing order based on their weight.

Initialize the MST: Start with an empty MST. The MST will be gradually constructed by adding edges one by one.

Add Edges: Iteratively add the next smallest edge to the MST, provided that it does not form a cycle with the edges already included. This step uses a disjoint-set data structure (also known as union-find) to efficiently manage and detect cycles.

Repeat Until Complete: Continue this process until the MST contains exactly V−1 edges, where V is the number of vertices in the graph.

Greediness of Kruskal's Algorithm

Kruskal's algorithm is considered greedy because, at each step, it chooses the smallest available edge that does not violate the MST properties (specifically, it does not form a cycle). This local optimization (choosing the smallest edge) leads to a global optimization (the MST) because the sum of the edges' weights is minimized.

Example

Let's consider a graph with vertices A, B,C, D,E and the following edges with weights:

(A,B)=1

(A,C)=5

(A,D)=3

(B,D)=4

(B,E)=2

(C,E)=7

(D,E)=6

Step-by-Step Execution:

Sort Edges:

(A,B)=1

(B,E)=2

(A,D)=3

(B,D)=4

(A,C)=5

(D,E)=6

(C,E)=7

Initialize the MST: Initially empty.

Add Edges:

Add (A,B)=1 (smallest edge, no cycle)

Add (B,E)=2 (next smallest, no cycle)

Add (A,D)=3 (next smallest, no cycle)

Add (B,D)=4 (next smallest, no cycle formed)

Skip (A,C)=5, (D,E)=6, and (C,E)=7 as adding any of them would form a cycle.

The MST now includes the edges (A,B)=1 (B,E)=2, (A,D)=3, and (B,D)=4 connecting all vertices with the minimum total weight of 10.

References

Shaffer, C. A. (2012). *Data structures and algorithm analysis*. Dover Publications.

326 words

Permalink Show parent

Re: Week 4

by Moustafa Hazeen - Tuesday, 16 July 2024, 3:25 AM

Your explanation of Kruskal's Algorithm is very clear and well-organized, covering both the algorithm's steps and its greedy nature effectively. The example is accurate and helps illustrate the process. To improve, you could provide a bit more detail about the Union-Find data structure and double-check the APA citation format for completeness. Excellent work!

53 words

Permalink Show parent

Re: Week 4

by Wingsoflord Ngilazi - Thursday, 18 July 2024, 1:04 AM

This is an excellent piece of work. Your explanations are very clear and concise. Your examples demonstrate an excellent grasp of Kruskal's Algorithm. Keep up the good work.

28 words

Permalink Show parent

Re: Week 4

by Siraajuddeen Adeitan Abdulfattah - Monday, 15 July 2024, 4:43 PM

Kruskal's algorithm is used to find Minimum Spanning Tree (MST) of a connected, undirected graph, where MST is a subset of edges connecting all vertices with minimum total edges possible and without any cycle. Kruskal's Algorithm is said to be greedy because it always chooses the next shortest edge that is not part of a cycle when building Minimum Spanning Tree (MST) (Ravikiran, 2024).

Kruskal's Algorithm functions following these steps:

It sorts all edges by the edge weight in an order of increase

It then selects the smallest or lowest edge

It verifies that the selected edge doesn't form a loop or form a cycle in spanning a tree
The selected edge is added to the MST, if doesn't belong in a cycle or loop. If it does, it is discarded
It selects the next lowest or smallest edge and repeat steps three and four, until the MST contains |V| - 1 edges (Ravikiran, 2024).

Example:

Let's look at a graph G with vertices (V) = {A, B, C, D, E, F} and edges (E) = {AB: 7, BC: 3, CD: 4, DE: 5, EF: 2, AC: 8, CF: 3, BF: 5, BD: 6, FD: 2}

We apply the Kruskal's Algorithm first step by sorting the edges in the ascending order based on the weights on the edges.

EF: 2

FD: 2

BC: 3

CF: 3

CD: 4

DE: 5

BF: 5

BD: 6

AB: 7

AC: 8

Second step is to choose the smallest edge EF with weight 2 and add it to the Minimum Spanning Tree (MST). As it is the first edge to be added in the MST, it will not form a cycle.

Third step choose the next smallest edge FD with weight 2 and add it to the MST.

Likewise, add edges BC with weight of 3 and CF with weight of 3 to the MST as both the edges do not form a cycle.

The next smallest edge is CD with weight 4. It should not be added into the MST because it will generate a cycle. So, the edge CD is discarded.

The next smallest edge is DE with weight 5. It should not be added into the MST because it creates a loop/cycle. So, the edge DE is dropped/discarded.

Likewise, the next smallest edge BF with weight 5 is discarded as it forms a cycle in the MST.

Choose the next smallest edge BD with weight 6. This edge will not be included in the MST as it creates a cycle. So, the edge BD is discarded.

Choose the next smallest edge AB with weight 7. This edge will be added to the MST as it does not create a cycle.

The final edge AC with weight 8 is discarded as it will form a cycle in the MST.

The algorithm ends because it has visited all the vertices in the graph G.
The Minimum Spanning Tree end result include six (6) vertices and five (5) edges listed below and seen in the image,

AB: 7

BC: 3

CF: 3

FD: 2

EF: 2

The cost of the minimum spanning tree is the sum of the weight of all edges in the MST. With the cost of 17.

Reference:

Rvikiran. A.S. (June, 2024). Your One-Stop Solution to learn Kruskal Algorithm from scratch. Retrieved from: https://www.simplilearn.com/tutorials/data-structure-tutorial/kruskal-algorithm

553 words

Permalink Show parent

Re: Week 4
by Fadi Al Rifai - Tuesday, 16 July 2024, 4:55 PM

Hi Siraajuddeen,
Your post was well-detailed about how Kruskal’s algorithm works, and I like your description of examples.
Keep it up.

21 words

Permalink Show parent

Re: Week 4

by Tousif Shahriar - Wednesday, 17 July 2024, 10:16 PM

Hello Siraajuddeen,
The step-by-step explanation is very clear! The visual representation helped me grasp the process of Kruskal's Algorithm and see how the MST is built over time.
Thanks for sharing!
31 words

Permalink Show parent

Re: Week 4

by Akomolafe Ifedayo - Monday, 15 July 2024, 4:43 PM

In Chapter 11 of the Shaffer text, we are introduced to Kruskal's Algorithm for determining a Minimum Spanning Tree within a graph. This algorithm is described as a greedy algorithm. Describe in your own words how Kruskal's algorithm works and be sure to articulate in your description what makes Kruskal's algorithm 'greedy'.
Include one or two examples to explain your thought process to show what is occurring and how the methodology works. Use APA citations and references for any sources used.

Kruskal's algorithm is a popular method used to find the Minimum Spanning Tree (MST) of a connected, undirected graph (GeeksforGeeks, 2023). The MST is a subset of the edges that connects all the vertices together without any cycles and with the minimum possible total edge weight.

Steps of Kruskal's Algorithm:

1. **Sort the edges**: Begin by sorting all the edges of the graph in non-decreasing order based on their weights.

2. **Initialize the MST**: Start with an empty MST, which will ultimately contain exactly V−1 edges if there are V vertices in the graph.

3. **Add edges to the MST**:

- Pick the smallest edge from the sorted list.
- Check if adding this edge to the MST will form a cycle.
- If no cycle is formed, add this edge to the MST.
- Repeat until the MST contains V−1 edges.

Kruskal's algorithm is considered "greedy" because it makes a series of choices, each of which looks the best at the moment. Specifically, it always picks the smallest edge available that doesn't form a cycle, aiming to build the MST incrementally so that each step is locally optimal (Shaffer, 2011).

Example 1:

Consider a graph with vertices A, B, C, and D, and edges with weights as follows:

- AB: 1
- AC: 4
- AD: 3
- BC: 2
- BD: 5
- CD: 6

Steps:

1. Sort edges: AB (1), BC (2), AD (3), AC (4), BD (5), CD (6).
2. Start with an empty MST.
3. Add AB (1) to MST.
4. Add BC (2) to MST.
5. Add AD (3) to MST (does not form a cycle).
6. Stop as the MST now has V−1=3 edges.

Example 2:

Consider a graph with vertices P, Q, R, and S, and edges with weights:

- PQ: 2
- PR: 3
- PS: 1
- QR: 4
- QS: 5
- RS: 6

Steps:

1. Sort edges: PS (1), PQ (2), PR (3), QR (4), QS (5), RS (6).
2. Start with an empty MST.
3. Add PS (1) to MST.
4. Add PQ (2) to MST.
5. Add PR (3) to MST (does not form a cycle).
6. Stop as the MST now has V−1=3 edges.

In both examples, the algorithm continually selects the smallest edge that does not create a cycle, ensuring the total weight of the MST is minimized, illustrating its greedy approach.

References

GeeksforGeeks. (2023). Kruskal's Minimum Spanning Tree (MST) Algorithm. https://www.geeksforgeeks.org/kruskals-minimum-spanning-tree-algorithm-greedy-algo-2/

Shaffer, C., A. (2011). A Practical Introduction to Data Structures and Algorithm Analysis. http://www.cse.ust.hk/~dekai/271/notes/L08/L08.pdf

503 words

Permalink Show parent

Re: Week 4
by Mahmud Hossain Sushmoy - Tuesday, 16 July 2024, 10:04 PM

Hello Akomolafe,
Thank you for your post. Your explanation of Kruskal's Algorithm is thorough and well-structured, clearly outlining the steps involved in the process and why it is considered a greedy algorithm. The examples you provided effectively illustrate the algorithm in action, making it easier to understand how it works and why it selects the smallest edges while avoiding cycles. Including citations and references enhances the credibility of your explanation. Overall, you have successfully conveyed the essence of Kruskal's Algorithm and its greedy nature.
84 words

Permalink Show parent

Re: Week 4
by Manahil Farrukh Siddiqui - Tuesday, 16 July 2024, 9:38 PM

Kruskal's Algorithm is a fundamental approach used to identify the Minimum Spanning Tree (MST) in a weighted graph by employing a greedy strategy (Cifci, 2023). The algorithm proceeds through a series of well-defined steps aimed at minimizing the overall weight of the spanning tree while ensuring that all vertices are connected.

The algorithm starts by selecting the lightest edges in the graph (Simplilearn, 2021). Imagine a network of cities linked by roads, each with a different cost. The first step is to gather all these costs and sort them in ascending order. This prioritization of the least expensive connections helps lay the foundation for constructing a cost-effective MST.

As Kruskal's Algorithm progresses, it adds the least costly edges one by one, ensuring no cycles are formed. Each edge is evaluated to check if its inclusion would create a cycle within the growing network (GeeksforGeeks, 2012). If the addition of an edge does not form a cycle, it is included in the MST, connecting the relevant vertices.

The algorithm continues this process until all vertices are connected, focusing on using the minimum number of edges. This approach ensures that the resulting MST spans the entire graph while keeping the total connection cost as low as possible (Simplilearn, 2021).

Kruskal's Algorithm is characterized by its greedy nature, always choosing the least expensive edge available. This strategy involves making the most cost-effective choice at each step without considering future implications. Although this method may seem short-sighted, it is effective as long as cycles are avoided, ultimately leading to the optimal MST (Cifci, 2023). By consistently selecting the cheapest edges, Kruskal's Algorithm efficiently constructs an MST that balances cost and connectivity.

Example

Consider a graph with cities A, B, C, and D connected by roads with the following costs: A-B = $10, A-C = $15, B-C = $8, B-D = $20, and C-D = $12.

Applying Kruskal's Algorithm:

1. Sort the edges: B-C ($8), A-B ($10), C-D ($12), A-C ($15), B-D ($20).

2. Add B-C since it is the cheapest and does not form a cycle.

3. Add A-B as it connects to B without forming a cycle.

4. Skip A-C because it would create a cycle (A-B-C).

5. Add C-D as it connects C and D without forming a cycle.

6. Skip B-D because it is the most expensive and forms a cycle (B-C-D-B).

The resulting MST includes the edges B-C, A-B, and C-D, with a total cost of $30. This example demonstrates how Kruskal's Algorithm effectively prioritizes low-cost connections and avoids cycles to construct an MST that optimally balances cost and connectivity.

References

Cifci, M. A. (2023, March 8). Kruskal's algorithm. Medium. https://themanoftalent.medium.com/kruskals-algorithm-11f3229138c6

GeeksforGeeks. (2012, October 30). Kruskal's minimum spanning tree algorithm | greedy algo-2 - geeksforgeeks. GeeksforGeeks. https://www.geeksforgeeks.org/kruskals-minimum-spanning-tree-algorithm-greedy-algo-2/

Simplilearn. (2021, July 29). Kruskal Algorithm: Overview & Create Minimum Spanning Tree | Simplilearn. Simplilearn.com. https://www.simplilearn.com/tutorials/data-structure-tutorial/kruskal-algorithm

473 words

Permalink Show parent

Re: Week 4

by Mahmud Hossain Sushmoy - Tuesday, 16 July 2024, 10:06 PM

Hello Manahil,
Your explanation of Kruskal's Algorithm is clear and insightful, providing a step-by-step breakdown of the process and effectively conveying its greedy nature. The real-world analogy of cities and roads helps to contextualize the algorithm, making it more relatable and easier to grasp. Thank you for your contribution to the discussion forum this week.

55 words

Permalink Show parent

Re: Week 4

by Muritala Akinyemi Adewale - Tuesday, 16 July 2024, 9:39 PM

Understanding Kruskal's Algorithm: A Greedy Approach to Minimum Spanning Trees

Kruskal's algorithm, introduced in Chapter 11 of Shaffer (2011), is a greedy algorithm for finding the minimum spanning tree (MST) of a weighted, connected graph. Here's a breakdown of how it works and why it's considered greedy:

Core Idea:

Kruskal's algorithm builds the MST in an incremental fashion, focusing on adding the "cheapest" (lowest weight) edges at each step. It's like building a bridge network across islands, starting with the shortest, most cost-effective connections first.

The Greedy Approach:

The "greedy" aspect of Kruskal's algorithm lies in its decision-making process. At each step, it chooses the edge with the minimum weight, **regardless of its potential impact on the overall structure of the MST.** This might seem counterintuitive, but it ultimately leads to finding the MST in most cases.

Steps:

1. **Sort Edges by Weight:** Arrange all edges in the graph from the lowest weight (cheapest) to the highest weight (most expensive).
2. **Initialize Forest:** Create a forest (collection of unconnected trees) where each vertex is a separate tree.

3. **Iterate Through Edges:** Process the sorted edges one by one.
    - **Check for Cycle Creation:** If adding the current edge would create a cycle (connecting two already connected trees), discard it.
    - **Merge Trees:** If adding the current edge doesn't create a cycle, merge the two trees that the edge connects. This essentially adds an edge to the growing MST.
4. **Continue Until MST is Formed:** Repeat step 3 until all vertices are connected in a single tree (the MST), and no more edges can be added without creating a cycle.

Example:

Consider a graph with the following edges and weights:

- A-B (weight 3)
- B-C (weight 2)
- C-D (weight 4)
- A-C (weight 1)
- D-E (weight 5)
- B-D (weight 6)

Solution:

1. Sort edges: A-C (1), B-C (2), A-B (3), C-D (4), B-D (6), D-E (5)
2. Initialize forest: Each vertex (A, B, C, D, E) is a separate tree.
3. Process edges:
    - A-C (weight 1): Doesn't create a cycle, so merge trees containing A and C. Now we have two trees: {A, C} and {B, D, E}.
    - B-C (weight 2): Connects existing connected vertices (B and C), so discard.
    - A-B (weight 3): Doesn't create a cycle, so merge trees containing A and B with the tree containing C. Now we have one tree: {A, B, C, D, E} (the MST).
4. Stop: All vertices are connected in a single tree, and no more edges can be added without creating a cycle.

Why is it Greedy?

Kruskal's algorithm focuses on the immediate benefit of adding the cheapest edge at each step. It doesn't necessarily consider the long-term implications of that choice on the overall structure of the MST. However, by prioritizing the cheapest edges first, it often leads to the formation of the MST with the minimum total weight in most practical scenarios.

Reference:

Shaffer, C. A. (2011). Data structures and algorithm analysis in C++ (4th ed.). Dover Publications. (Chapter 11: Graph Algorithms)

488 words

Permalink Show parent

Re: Week 4

by Mahmud Hossain Sushmoy - Tuesday, 16 July 2024, 10:09 PM

Hello Muritala,
Your explanation of Kruskal's Algorithm is clear and concise, effectively highlighting the key steps and the rationale behind its greedy approach. The use of an example with specific edge weights helps to concretize the concept, making the process more understandable. Overall, the explanation successfully conveys the core principles and functionality of Kruskal's Algorithm. Thank you for your contribution to the discussion forum this week.

66 words

Permalink Show parent

Re: Week 4

by Tyler Huffman - Tuesday, 16 July 2024, 10:56 PM

Kruskal’s algorithm

Kruskal’s algorithm can be used to find the minimum spanning tree, or MST, of a given graph; as defined by Geeksforgeeks, an MST is the spanning tree that has the minimum weight among all possible spanning trees (2024). In other words, a subgraph that is an acyclic tree and covers all vertices while minimizing cost. The MST will have an edge count of V - 1 (E = V - 1). The uses for an algorithm that calculates optimal solutions like this are vast and therefore understanding it is of significant importance.

How does it work?

The algorithm begins with some initialization, setting up essential auxiliary data structures. First, a set of all vertices and a set of all edges should be made. Knowing the total amount of vertices allows us to easily determine when we are done processing. The set of edges (E) is sorted by weight, lowest to highest. Lastly we create a set of ‘safe’ edges (we will refer to this set as T) that will ultimately be the structure for our minimum spanning tree, our desired output. This is of course empty when initialized. A safe edge in this context means that, if added to our MST, it would not create a cycle.

We begin forming our MST by taking the lowest cost safe edge from E; during the first pass, this will simply be the first edge in E because our set is sorted and no cycles are possible at this point. We add this edge to T. We then do it again: go into E, take the next lowest weighted edge, and check it for safeness. If the edge is safe, add it to T. If not, discard it and continue the process with the next edge in E. As stated above, our final MST will have an amount of edges equal to V - 1. So long as we have not met this requirement, we will continue this process. When finished, our set T of safe edges will indeed be an MST of the original input graph.

What makes it greedy?

The ‘greediness’ of the algorithm is evident. Greedy algorithms are “algorithms that make locally optimal choices at each step with the hope of finding a global optimum solution” (Geeksforgeeks, 2024). It is clear how Kruskal’s algorithm fits within this class. At each ‘step’ of the process, the best local option option is selected. Best in this case meaning the edge with the lowest cost. We continue using this greedy method until arriving at an optimal global solution.

References

Geeksforgeeks. (2024, March 8). *What is Minimum Spanning Tree (MST).* https://www.geeksforgeeks.org/what-is-minimum-spanning-tree-mst/

Geeksforgeeks. (2024, May 2). *Greedy Algorithms.* https://www.geeksforgeeks.org/greedy-algorithms/

446 words

Permalink Show parent

Re: Week 4

by Romana Riyaz (Instructor) - Wednesday, 17 July 2024, 12:31 AM

Tyler,
Thank you for your response. The explanation of Kruskal’s algorithm effectively outlines its purpose, process, and greedy nature. Starting with a clear definition of the Minimum Spanning Tree (MST) and its significance provides a solid foundation. The step-by-step breakdown of the algorithm's initialization, edge selection, and cycle avoidance is thorough and easy to follow. The discussion on the algorithm's "greediness" is well-explained, highlighting how locally optimal choices lead to a global optimum solution. Additionally, referencing authoritative sources like Geeksforgeeks adds credibility and offers readers a path for further reading. Overall, this explanation is comprehensive and well-structured, making the concept of Kruskal’s algorithm accessible and understandable.
Best,
Romana

108 words

Permalink Show parent

Re: Week 4

by Anthony Jones - Wednesday, 17 July 2024, 2:08 AM

Hello,

Good post! How can we check if the edge would make a cycle? what is the time complexity of checking? It's good to understand how we can be sure that Kruskal's algorithm creates the MST. How do we know that the edge that makes a cycle shouldn't be in the tree and isn't a better option than the ones we added?

God bless!

Anthony

65 words

Permalink Show parent

Re: Week 4

by Winston Anderson - Wednesday, 17 July 2024, 9:23 PM

Hi Tyler,
This is a clear and structured explanation of Kruskal's algorithm. You've done a commendable job explaining the algorithm's operational details and the rationale behind its classification as a greedy algorithm. Your text is well-organized, with distinct sections for different aspects of the algorithm, which aids in understanding.

49 words

Permalink Show parent

Re: Week 4

by Tousif Shahriar - Wednesday, 17 July 2024, 10:19 PM

Hello Tyler,
Excellent overview of Kruskal's algorithm and its greedy nature! The explanation is clear and thorough regarding the initialization and process of Kruskal's algorithm. It would be interesting to elaborate on a few real-world applications where Kruskal's algorithm is particularly useful.
Thanks for sharing!

45 words

Permalink Show parent

Re: Week 4

by Wingsoflord Ngilazi - Wednesday, 17 July 2024, 12:11 AM

Describe in your own words how Kruskal's algorithm works and be sure to articulate in your description what makes Kruskal's algorithm 'greedy'.

Include one or two examples to explain your thought process to show what is occurring and how the methodology works.

Kruskal's Algorithm is classified as a greedy algorithm. It is used to find the Minimum Spanning Tree (MST) of a connected, undirected graph. An MST can be defined as a subset of the edges in a graph that connects all vertices with the minimum possible total edge weight without forming any cycles (Shaffer, 2013).

According to Shaffer (2013), the Kruskal's Algorithm follows the steps below:

i. Start by sorting all edges: Initially, sort all the edges in the graph by their weights in ascending order.

ii. Get started with the MST: Create a forest (a collection of trees), where each vertex in the graph is its own individual tree.

iii. Edge Selection: It is important to go through the sorted list of edges and for each edge:

a. Check if adding the edge to the MST would form a cycle.

b. If no cycle is formed, add the edge to the MST.

c. If a cycle is formed, leave out the edge.

iv. Repeat: Repeat step 3 until the forest becomes a single tree.

v. The algorithm stops when the vertices have been connected to become a single tree.

The Kruskal's Algorithm is classified as greedy since it makes a series of locally optimal choices with the aim of finding a global optimum. In this scenario, the locally optimal choice is always selecting the smallest edge that does not form a cycle.

Let's consider a graph with the following features:

Vertices = {A, B, C, D}

Edges and their respective weights = (A-B, 1), (B-C, 3), (C-D, 4), (A-D, 2), (A-C, 5)

Bellow is how the Kruskal's Algorithm will operate:

i. Sort edges: (A-B, 1), (A-D, 2), (B-C, 3), (C-D, 4), (A-C, 5)

ii. Initialize forest: {A}, {B}, {C}, {D}

iii. Edge Selection:

a. Add (A-B, 1): Forest becomes {A, B}, {C}, {D}

b. Add (A-D, 2): Forest becomes {A, B, D}, {C}

c. Add (B-C, 3): Forest becomes {A, B, C, D}

d. Edge (C-D, 4) and (A-C, 5) are not added because the MST is complete with 3 edges.

Since the algorithm works by selecting the least expensive edges, the MST will be as follows: Edges (A-B, 1), (A-D, 2), (B-C, 3) with total weight = 1 + 2 + 3 = 6.

References

Shaffer, C. A. (2013). A Practical Introduction to Data Structures and Algorithm Analysis. Dover Publications.

433 words

Permalink Show parent

Re: Week 4
by Mejbaul Mubin - Wednesday, 17 July 2024, 4:46 AM

Hi Wingsoflord Ngilazi,

Your explanation of Kruskal's algorithm is thorough and well-organized. You effectively illustrate its greedy nature by emphasizing the selection of smallest edges without forming cycles, leading to a minimum spanning tree. Your example further clarifies the algorithm's steps and outcome. Well done!
45 words

Permalink Show parent

Re: Week 4
by Christopher Mccammon - Wednesday, 17 July 2024, 8:08 PM

Hi Wingsoflord Ngilazi
Your explanation of Kruskal's algorithm is well-crafted and demonstrates a solid understanding of the topic. You have effectively outlined the algorithm's steps and provided a clear example that illustrates how it works in practice. The structure and organization of your explanation are logical and easy to follow. Good job!
52 words

Permalink Show parent

Re: Week 4
by Tousif Shahriar - Wednesday, 17 July 2024, 10:21 PM

Hello Wingsoflord Ngilazi,
Excellent walkthrough of the edge selection process! To enhance the explanation, you could mention the use of the union-find data structure (disjoint-set) for efficiently managing the forest and checking for cycles.
Thanks for sharing!
37 words

Permalink Show parent

Re: Week 4
by Nour Jamaluddin - Wednesday, 17 July 2024, 3:59 AM

Kruskal's algorithm is a method used to find the minimum spanning tree (MST) of a connected, weighted, undirected graph. A minimum spanning tree is a subset of the edges that connects all vertices together without any cycles and has the minimum total edge weight (GeeksforGeeks, 2023).

- **What makes Kruskal's algorithm 'greedy'**

Kruskal's algorithm is considered greedy because at each step, it picks the smallest edge that does not form a cycle with the previously selected edges.

- **How Kruskal's algorithm works**

First, all edges of the graph are sorted in non-descending order of their weights. A subset is then created for each vertex, initializing each vertex as its own parent as a separate tree to be the union-find structure. It iterates over the sorted edges and for each edge, it is best to check whether adding them to the MST will form a cycle using a union-find structure.

Finally, Kruskal's algorithm efficiently manages the MST construction process by sorting edges and using a union-find structure.

- **Example:**

Consider a graph with vertices A, B, C, D, and E, and the following edges with their weights:

AB: 2

AC: 3

BC: 4

BD: 5

CD: 1

DE: 6

Sort edges: CD (1), AB (2), AC (3), BC (4), BD (5), DE (6)

Initialize: Five separate trees: A, B, C, D, E

Build MST:

Add CD (weight 1): Trees: A, B, C-D, E

Add AB (weight 2): Trees: A-B, C-D, E

Add AC (weight 3): Creates a cycle, discard.

Add BC (weight 4): Creates a cycle, discard.

Add BD (weight 5): Trees: A-B, C-D, E-B

Add DE (weight 6): Creates a cycle, discard.

The MST consists of edges CD, AB, and BD with a total weight of 1 + 2 + 5 = 8.

References

GeeksforGeeks. 2023. *Kruskal's Minimum Spanning Tree (MST) Algorithm*. https://www.geeksforgeeks.org/kruskals-minimum-spanning-tree-algorithm-greedy-algo-2/

300 words

Permalink Show parent

Re: Week 4

by Mejbaul Mubin - Wednesday, 17 July 2024, 4:45 AM

Hi Nour Jamaluddin,

Your explanation of Kruskal's algorithm for finding the minimum spanning tree is clear and well-structured. You effectively outline its greedy nature and step-by-step process, supported by a concise example. Well done!

34 words

Permalink Show parent

Re: Week 4

by Christopher Mccammon - Wednesday, 17 July 2024, 8:02 PM

Hi Nour Jamaluddin
You have a solid understanding of Kruskal's algorithm. Your explanation of Kruskal's algorithm is clear and provides a good foundational understanding. The basic steps and an example help in illustrating how the algorithm works. Good job!

39 words

Permalink Show parent

Re: Week 4

by Winston Anderson - Wednesday, 17 July 2024, 8:51 AM

Kruskal's Algorithm: A Greedy Approach to Finding a Minimum Spanning Tree

Kruskal's algorithm is a well-known method for finding the Minimum Spanning Tree (MST) of a connected, weighted, undirected graph. This algorithm is classified as a greedy algorithm because it makes a series of locally optimal choices with the hope of finding a global optimum solution. Below, we will detail how Kruskal's algorithm works and explain its greedy nature with examples.

How Kruskal's Algorithm Works

The primary goal of Kruskal's algorithm is to construct a spanning tree with the minimum possible total edge weight. The algorithm follows these steps:

1. Sort the Edges: Begin by sorting all the edges of the graph in non-decreasing order of their weights.

2. Initialize the MST: Start with an empty spanning tree.

3. Add Edges: Iterate through the sorted edges and add each edge to the spanning tree if it does not form a cycle with the edges already included in the tree. This step typically involves using the Union-Find data structure to efficiently detect cycles.

4. Repeat: Continue adding edges until the spanning tree contains V-1 edges, where V is the number of vertices in the graph (GeeksforGeeks, 2023; Javatpoint, n.d.).

Why Kruskal's Algorithm is Greedy

Kruskal's algorithm is considered greedy because at each step, it selects the smallest available edge that does not form a cycle. This local optimum choice is made with the expectation that it will lead to the global optimum solution, which is the MST. The algorithm does not reconsider its choices once made, which is a hallmark of greedy strategies (Board Infinity, n.d.).

Example 1: Simple Graph

Consider a graph with four vertices and five edges with the following weights:

- Edge (0, 1) with weight 10

- Edge (0, 2) with weight 6

- Edge (0, 3) with weight 5

- Edge (1, 3) with weight 15

- Edge (2, 3) with weight 4

Step-by-Step Execution:

1. Sort the edges by weight: (2, 3), (0, 3), (0, 2), (0, 1), (1, 3)

2. Initialize MST: Empty set

3. Add edges:

   - Add edge (2, 3) with weight 4 (no cycle)

   - Add edge (0, 3) with weight 5 (no cycle)

   - Add edge (0, 2) with weight 6 (no cycle)

   - Skip edge (0, 1) with weight 10 (would form a cycle)

   - Skip edge (1, 3) with weight 15 (would form a cycle)

The resulting MST includes edges (2, 3), (0, 3), and (0, 2) with a total weight of 15.

Example 2: Complex Graph

Consider a graph with six vertices and nine edges with the following weights:

- Edge (0, 1) with weight 4
- Edge (0, 2) with weight 4
- Edge (1, 2) with weight 2
- Edge (1, 3) with weight 5
- Edge (2, 3) with weight 3
- Edge (2, 4) with weight 4
- Edge (3, 4) with weight 7
- Edge (3, 5) with weight 6
- Edge (4, 5) with weight 1

Step-by-Step Execution:

1. Sort the edges by weight: (4, 5), (1, 2), (2, 3), (0, 1), (0, 2), (2, 4), (1, 3), (3, 5), (3, 4)

2. Initialize MST: Empty set

3. Add edges:

   - Add edge (4, 5) with weight 1 (no cycle)
   - Add edge (1, 2) with weight 2 (no cycle)
   - Add edge (2, 3) with weight 3 (no cycle)
   - Add edge (0, 1) with weight 4 (no cycle)
   - Add edge (0, 2) with weight 4 (no cycle)
   - Skip edge (2, 4) with weight 4 (would form a cycle)
   - Skip edge (1, 3) with weight 5 (would form a cycle)
   - Skip edge (3, 5) with weight 6 (would form a cycle)
   - Skip edge (3, 4) with weight 7 (would form a cycle)

The resulting MST includes edges (4, 5), (1, 2), (2, 3), (0, 1), and (0, 2) with a total weight of 14.

Conclusion

Kruskal's algorithm is a powerful and efficient method for finding the Minimum Spanning Tree of a graph. Its greedy nature ensures that at each step, the smallest available edge is chosen, leading to an overall optimal solution. By understanding the algorithm's steps and seeing it in action through examples, one can appreciate how local decisions can effectively lead to a globally optimal result.

References:

Board Infinity. (n.d.). *Working of Kruskal's Algorithm.* https://www.boardinfinity.com/blog/kruskal-algorithm/

GeeksforGeeks. (2023, October 5). *Kruskal s Minimum Spanning Tree (MST) Algorithm*. GeeksforGeeks. https://www.geeksforgeeks.org/kruskals-minimum-spanning-tree-algorithm-greedy-algo-2/

Javatpoint. (n.d.). *Kruskal's Minimum Spanning Tree Algorithm - javatpoint*. www.javatpoint.com. https://www.javatpoint.com/kruskals-minimum-spanning-tree-algorithm

746 words

Permalink Show parent

Re: Week 4

by Christopher Mccammon - Wednesday, 17 July 2024, 7:53 PM

Hi Winston
Your explanation of Kruskal's algorithm is thorough and well-crafted. You have successfully broken down a complex algorithm into clear, manageable steps that are easy to follow. The inclusion of detailed examples effectively illustrates how the algorithm works in practice, which enhances understanding. The explanation of the algorithm's greedy nature is particularly well-done, providing insight into why Kruskal's algorithm is effective. Good job!

64 words

Permalink Show parent

Re: Week 4

by Muritala Akinyemi Adewale - Wednesday, 17 July 2024, 11:04 PM

This is a well-crafted explanation of dynamic programming using the knapsack problem. You've clearly outlined the problem and demonstrated how to build the solution using a table to store intermediate results. Your structured approach and step-by-step explanation make it very comprehensible. Excellent work!

43 words

Permalink Show parent

Re: Week 4

by Christopher Mccammon - Wednesday, 17 July 2024, 7:05 PM

Kruskals Algorithm is a technique used to discover a Minimum Spanning Tree (MST) within a graph. An MST is a set of edges that links all vertices without creating cycles. With the total edge weight possible. This method follows an approach making decisions that appear optimal at each step, in the hope of reaching the solution(Cormen et al,2009).

To begin Kruskals Algorithm all edges in the graph are sorted in ascending order based on their weights. This sorting ensures that the algorithm always considers the edge. Subsequently an empty graph is initialized to hold the MST edges(Cormen et al,2009).

The algorithm then loops through the edge list. For each edge it verifies if adding it to the MST would generate a cycle. To efficiently manage cycle detection a union find data structure is employed to monitor vertices. If adding an edge does not create a cycle it becomes part of the MST. This process repeats until the MST contains edges, where V represents the number of vertices, in the graph indicating completion of building the MST(Cormen et al,2009).

The algorithm is often described as greedy because it makes decisions locally that seem best at each step. It consistently picks the edge that doesn't form a loop aiming to reduce the weight of the spanning tree bit, by bit. This choice is made without considering the structure of the graph; instead it relies on the idea that choosing the edge will eventually lead to an optimal outcome. By following this approach Kruskal's Algorithm efficiently constructs the Minimum Spanning Tree (MST) while ensuring that the overall weight stays as low as possible. The algorithms greedines is evident in how it sellects edges based on their weights(GeeksforGeeks,2021).

Reference:
Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). Introduction to algorithms (3rd ed.). The MIT Press. Retrieved

from https://mitpress.mit.edu/books/introduction-algorithms

GeeksforGeeks. (2021). Kruskal’s algorithm for finding Minimum Spanning Tree. https://www.geeksforgeeks.org/kruskals-algorithm-using-union-find-algorithm/

319 words

Permalink Show parent

Re: Week 4

by Winston Anderson - Wednesday, 17 July 2024, 9:20 PM

Hi Christopher,
This explanation provides a basic overview of Kruskal's Algorithm and correctly identifies it as a greedy algorithm used for finding a Minimum Spanning Tree (MST) in a graph. The use of references is good practice, lending credibility to the description.

42 words

Permalink Show parent

Re: Week 4

by Tousif Shahriar - Wednesday, 17 July 2024, 10:33 PM

Hello Christopher,
Great explanation of Kruskal's Algorithm! It would be clearer to say that the MST connects all vertices with the minimum total edge weight possible. You've mentioned the use of a union-find data structure for cycle detection, which is crucial for the algorithm. Also, your explanation of the greedy nature of Kruskal's Algorithm is spot-on.
Thanks for sharing!

59 words

Permalink Show parent

Re: Week 4

by Wingsoflord Ngilazi - Thursday, 18 July 2024, 12:53 AM

Hi Christopher,
Thank you for sharing this well-written post. Your explanations are clear and they demonstrate a thorough grasp of Kruskal's Algorithm. Keep up the excellent work.

27 words

Permalink Show parent

Re: Week 4

by Aye Aye Nyein - Wednesday, 17 July 2024, 8:07 PM

Kruskal's algorithm is a methodical approach to finding a Minimum Spanning Tree (MST) in a graph, where an MST is a subset of edges that connect all vertices with the minimum possible total edge weight without forming any cycles. It operates by systematically selecting the smallest edge available that connects two different components of the MST being constructed. This key characteristic classifies Kruskal's algorithm as a greedy algorithm.

Kruskal's Algorithm operates as follows:

1. Start by treating each vertex as its separate component.

2. Arrange all edges in the graph in ascending order based on their weights.

3. Sequentially examine each edge:

- Select the smallest edge that has not been included in the Minimum Spanning Tree (MST) yet.

- Verify that adding this edge will not create a cycle using a method like the Union-Find data structure.

4. Use the Union-Find data structure to check if connecting two vertices with the selected edge would form a cycle.

5. If adding the edge connects two different components (vertices from separate subsets), add it to the MST; otherwise, discard it.

6. Repeat steps until edges (where is the number of vertices) have been added to the MST.

7. The resultant set of edges constitutes the Minimum Spanning Tree of the graph.

Example:

Consider a graph with vertices and the following weighted edges:

-

-

-

-

-

-

-

Applying Kruskal's algorithm step by step:

1. Sort edges by weight:

-

-

-

-

-

-

-

2. Select edges:

- : Include because it forms no cycle.

- : Include because it forms no cycle.

- : Include because it forms no cycle.

- : Include because it forms no cycle.

The MST formed includes , , , and .

Greedy Nature of Kruskal’s Algorithm:

Kruskal’s algorithm is classified as greedy because it makes decisions locally (choosing the smallest edge available at each step) without considering the overall structure of the MST. The algorithm relies on the fact that selecting the smallest edge that does not cause a cycle at each step will eventually lead to the formation of the MST with the minimum total weight. By adhering to this greedy strategy, Kruskal’s algorithm ensures that it achieves an optimal solution for constructing the MST.

In summary, Kruskal’s algorithm’s greedy approach ensures that at every step, the algorithm makes the optimal choice that contributes towards minimizing the overall weight of the MST, thereby exemplifying its greedy nature.

Reference:

Shaffer, C. A. (2001). A Practical Introduction to Data Structures and Algorithm Analysis.

540 words

Permalink Show parent

Re: Week 4

by Winston Anderson - Wednesday, 17 July 2024, 9:13 PM

Hi Aye,
This is a well-structured and detailed explanation of Kruskal's algorithm. The description of the algorithm is clear, and the step-by-step process, along with the example, effectively illustrates how the algorithm works in practice. Additionally, the explanation of why Kruskal's algorithm is considered greedy is aptly supported with relevant details.
51 words

Permalink Show parent

Re: Week 4

by Muritala Akinyemi Adewale - Wednesday, 17 July 2024, 11:01 PM

Your explanation of the divide-and-conquer approach in merge sort is very clear and detailed. You've done a fantastic job of illustrating how the algorithm divides the array and merges the sorted subarrays. The logical flow and use of examples make it easy to understand. Well done!
46 words

Permalink Show parent

Re: Week 4

by Tousif Shahriar - Wednesday, 17 July 2024, 9:25 PM

In Chapter 11 of the Shaffer textbook, Kruskal’s Algorithm is introduced as a method for finding a Minimum Spanning Tree (MST) in a graph. A Minimum Spanning Tree is a subset of the edges in a connected, undirected graph that connects all the vertices together without any cycles and with the minimum possible total edge weight (Shaffer, 2010).

Kruskal’s Algorithm works in the following way:

- Sort all the edges in the graph in the non-decreasing order of their weights.
- Initialize an empty set to keep track of the edges included in the MST.

- Iterate through the sorted edge list and for each edge, also check if adding it to the MST would form a cycle.
  - If it does not form a cycle, add it to the MST.
  - If it forms a cycle, discard it.
- Repeat this process until the set contains exactly V−1 edges, where V is the number of vertices in the graph.

The algorithm uses a disjoint-set data structure (also known as a union-find data structure) to efficiently manage and check for cycles.

Kruskal's Algorithm can be considered greedy because it makes a series of choices that seem best at the moment. Such as at each step, it picks the smallest edge that does not form a cycle, and it does not consider the future consequences, which is greedy in nature. This local optimal choice leads to a globally optimal solution for the MST.

For example, we can consider a graph with 4 vertices and the following edges:

- Edge (A, B) with weight 1
- Edge (A, C) with weight 3
- Edge (B, C) with weight 2
- Edge (B, D) with weight 4
- Edge (C, D) with weight 5

The algorithm will work in the following way:

1. Sort edges by weight: (A, B), (B, C), (A, C), (B, D), (C, D)
2. Initialize MST: {}
3. Add edge (A, B): MST = {(A, B)}
4. Add edge (B, C): MST = {(A, B), (B, C)}
5. Add edge (A, C): Adding (A, C) forms a cycle, so discard it.
6. Add edge (B, D): MST = {(A, B), (B, C), (B, D)}

Hence, the MST is {(A, B), (B, C), (B, D)} with a total weight of 1 + 2 + 4 = 7

Reference

Shaffer, C. A. (2010). A Practical Introduction to Data Structures and Algorithm Analysis (3rd ed.). Retrieved from https://people.cs.vt.edu/~shaffer/Book/C++3e20100119.pdf

Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). Introduction to Algorithms (3rd ed.). MIT Press.

411 words

Permalink Show parent

Re: Week 4
by Muritala Akinyemi Adewale - Wednesday, 17 July 2024, 11:00 PM

This is an excellent breakdown of Dijkstra's algorithm. You have effectively explained each step and how the algorithm prioritizes the shortest path using a priority queue. The way you structured your explanation with clear examples helps in grasping the concept thoroughly. Great job!
43 words

Permalink Show parent

Re: Week 4
by Wingsoflord Ngilazi - Thursday, 18 July 2024, 12:59 AM

Hi Tousif,
I sincerely enjoyed reading your post. It clearly demonstrates a remarkable mastery of this week's learning content. Your explanations are very clear and your examples are relevant. Keep up the excellent work.
34 words

Permalink Show parent

Re: Week 4
by Loubna Hussien - Wednesday, 17 July 2024, 11:42 PM

Dicsussion unit 4:

Kruskal's algorithm is employed to find the shortest path in a connected weighted network. A spanning tree is a subgraph of the indirectly connected graph, and the minimum spanning tree is the one with the smallest sum of edge weights, where the weight of the tree is the total of these weights. The algorithm's primary goal is to identify a subset of edges that allows traversal of each vertex in the graph. It operates greedily, focusing on optimal solutions at each step rather than globally optimal ones (javatpoint, n.d.).

The steps for Kruskal's algorithm can be broken down as follows: First, sort all edges in ascending order based on their weights. Next, select the smallest edge and ensure it forms no cycles with the growing spanning tree. If no cycle is formed, the edge is added to the tree; otherwise, it is discarded. The Union-Find method is used to detect cycles during this process. Finally, repeat step 2 until the spanning tree has V−1V - 1V−1 edges.

Kruskal's algorithm exemplifies a greedy approach as it constructs the structure incrementally. It begins at a vertex and examines all possible edges to neighboring vertices, selecting the lowest-cost edge. In essence, the greedy choice is to pick the edge with the smallest weight that does not form a cycle in the developing minimum spanning tree (University of the People, n.d.).

Using the example from the Kruskal's Minimum Spanning Tree (MST) Algorithm tutorial on GeeksforGeeks (2023), where the graph has 9 vertices and 14 edges, the minimum spanning tree will have 9−1=89 - 1 = 89−1=8 edges.

References:

javatpoint. (n.d.). *Kruskal's Algorithm.* www.javatpoint.com.

https://www.javatpoint.com/kruskal-algorithm

University of the People. (n.d.). *Learning Guide Unit 4.* https://my.uopeople.edu/mod/book/view.php?id=359138&chapterid=426209

GeeksforGeeks. (2023). Kruskal s Minimum Spanning Tree MST
Algorithm. *GeeksforGeeks*. https://www.geeksforgeeks.org/kruskals-minimum-spanning-tree-algorithm-greedy-algo-2/

294 words

Permalink Show parent

Re: Week 4
by Manahil Farrukh Siddiqui - Thursday, 18 July 2024, 3:46 AM

Hello Loubna,

Your answer is a well-structured and detailed explanation of Kruskal's algorithm. You have effectively illustrates how the algorithm works in practice. Additionally, the explanation of why Kruskal's algorithm is considered greedy is aptly supported with relevant details.
39 words

Permalink Show parent

Re: Week 4
by Jobert Cadiz - Thursday, 18 July 2024, 5:17 AM

Hi Loubna,
Your answer provides a clear and concise explanation of Kruskal's algorithm, effectively summarizing its purpose and methodology. It accurately describes the algorithm's goal of finding the minimum spanning tree (MST) by selecting edges based on weight while avoiding cycles. However, the statement that Kruskal's algorithm finds the "shortest path" in a network is misleading; it specifically finds a minimum spanning tree, not the shortest path between two nodes.
70 words

Permalink Show parent

Re: Week 4
by Chong-Wei Chiu - Thursday, 18 July 2024, 9:15 AM

Hello, Loubna Hussien. Thank you for sharing your point of view on this topic. You explain the concept of Kruskal's Algorithm in your post. Additionally, you present how to use this algorithm to find the minimum spanning tree step by step, and in each step, you use a graph to illustrate which edge you choose. Based on these points, I'm sure that you have a comprehensive understanding of this topic.

70 words

Permalink Show parent

CS 3304 Discussion Forum Unit 4

by Jerome Bennett - Thursday, 18 July 2024, 3:14 AM

CS 3304 Discussion Forum Unit 4

Kruskal's algorithm is a method used to find the minimum spanning tree (MST) of a connected, undirected graph (Shaffer, 2011). The MST is a subset of the edges that connects all the vertices together, without any cycles, and with the minimum possible total edge weight.

Here is a summary of how Kruskal's algorithm works:

Sort all the edges: Start by sorting all the edges in the graph by their weight in ascending order.

Initialize the MST: Create a new graph (or data structure) that will hold the edges of the MST.

Add edges: Add the smallest edge to the MST that doesn't form a cycle. Continue adding the next smallest edge, ensuring that each new edge doesn't form a cycle with the edges already in the MST.

Repeat until the MST is complete: Keep adding edges until the MST contains exactly

V−1 edges, where V is the number of vertices in the original graph.

The algorithm is called 'greedy' because at each step, it chooses the smallest possible edge that can be added to the MST without forming a cycle (MIT OpenCourseWare, 2009). This local optimal choice (selecting the smallest edge) leads to a globally optimal solution (the minimum spanning tree) (Dasgupta et al., 2006).

For example, let's consider a graph with 5 vertices (P, Q, R, S, T) and 7 edges with the following weights:

P-Q: 1

P-R: 4

Q-R: 2

Q-S: 3

R-S: 5

R-T: 6

S-T: 7

Here's how Kruskal's algorithm would work on this graph (Shaffer, 2011):

Sort the edges by weight:

P-Q: 1

Q-R: 2

Q-S: 3

P-R: 4

R-S: 5

R-T: 6

S-T: 7

Initialize the MST: Start with an empty set for the MST.

Add the smallest edge:

Add P-Q (weight 1). MST now contains {P-Q}.

Add the next smallest edge:

Add Q-R (weight 2). MST now contains {P-Q, Q-R}.

Add the next smallest edge that doesn't form a cycle:

Add Q-S (weight 3). MST now contains {P-Q, Q-R, Q-S}.

Add the next smallest edge that doesn't form a cycle:

Add P-R (weight 4). But wait, adding P-R would form a cycle with P-Q and Q-R. So, we skip this edge.

Next, add R-S (weight 5). But adding R-S would form a cycle with Q-R and Q-S. So, we skip this edge.

Next, add R-T (weight 6). MST now contains {P-Q, Q-R, Q-S, R-T}.

Stop when the MST contains V-1 edges:

We have 5 vertices, so the MST needs 4 edges. Our MST is complete with edges {P-Q, Q-R, Q-S, R-T}.

```
   1     2     5
P ----- Q ----- R ----- S
 \     / \     / \     /
  \   /   \   /   \   /
   \ /     \ /     \ /
    R       S       T
        3       6
```

Final MST:

```
   1     2
P ----- Q ----- R ----- T
        |
        3
        |
        S
```

Kruskal’s algorithm successfully found the minimum spanning tree by greedily adding the smallest edge that didn’t form a cycle, ultimately connecting all vertices with the minimal total edge weight (Shaffer, 2011; MIT OpenCourseWare, 2009).

Reference

Dasgupta, S., Papadimitriou, C.H.,Vazirani, U.V. (2006). Chapter 4 Paths in Graphs in Algorithms. http://www.cs.berkeley.edu/~vazirani/algorithms/chap4.pdf

Shaffer, C. (2011). Lecture 8: Lecture 8: Kruskal’s MST Algorithm. http://www.cse.ust.hk/~dekai/271/notes/L08/L08.pdf

MIT OpenCourseWare. (2009, January 7). LEC 17 | MIT 6.046J / 18.410J Introduction to Algorithms (SMA 5503), Fall 2005 [Video]. YouTube.

Lec 17 | MIT 6.046J / 18.410J Introduc...

554 words

Permalink Show parent

Re: CS 3304 Discussion Forum Unit 4
by Manahil Farrukh Siddiqui - Thursday, 18 July 2024, 3:47 AM

Hello Jerome,

Your answer is well-structured. The step-by-step process, along with the example is clear and easy to understand. You have effectively illustrates how the algorithm works in practice. Additionally, the explanation of why Kruskal's algorithm is considered greedy is aptly supported with relevant details.

45 words

Permalink Show parent

Re: CS 3304 Discussion Forum Unit 4

by Jobert Cadiz - Thursday, 18 July 2024, 5:13 AM

Hi Jerome,
You have provided a comprehensive and informative response that effectively explains Kruskal's algorithm and its greedy nature. The example illustrates the process well, though a bit more detail in explaining the cycle-checking mechanism could enhance clarity. Overall, it's an excellent summary suitable for understanding Kruskal's algorithm. Keep up the great work!

53 words

Permalink Show parent

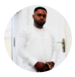
Re: CS 3304 Discussion Forum Unit 4

by Liliana Blanco - Thursday, 18 July 2024, 9:22 AM

Your explanation of Kruskal's Algorithm is really good! I really appreciate how you broke down the steps and used an example to show the process. Your point about the algorithm being 'greedy' because it always chooses the smallest edge aligns perfectly with my description. Your use of a practical example with specific vertices and edge weights provides a great explanation of how the algorithm avoids cycles and builds the MST.

70 words

Permalink Show parent



Re: Week 4

by Prince Ansah Owusu - Thursday, 18 July 2024, 3:56 AM

Kruskal's Algorithm for Minimum Spanning Tree

Kruskal's Algorithm is a fundamental method used to find a Minimum Spanning Tree (MST) in a weighted, connected, and undirected graph. It belongs to the category of greedy algorithms, known for making locally optimal choices to achieve a globally optimal solution.

How Kruskal's Algorithm Works:

1. **Initialization:**

- Begin with an empty list to store the edges of the MST.
- Sort all edges of the graph in the non-decreasing order of their weights.

Edge Selection:

- Iterate through the sorted list of edges.
- For each edge, check if adding it to the MST forms a cycle using a Union-Find data structure:
    - **Union-Find Operations:**
        - **Find:** Determine the root of the set containing a particular vertex.
        - **Union:** Merge two sets into one.
- If adding the edge does not form a cycle (i.e., the edge connects two different components), include it in the MST.

Completion:

- Continue adding edges until V−1V - 1V−1 edges have been added to the MST, where VVV is the number of vertices in the graph.

Greedy Nature of Kruskal's Algorithm:

Kruskal's Algorithm exhibits its greedy nature through its edge selection process:

- At each step, it selects the smallest available edge that does not form a cycle with the edges already included in the MST.
- By making these locally optimal choices, Kruskal's Algorithm ensures that the final MST is globally optimal, i.e., it has the minimum total weight among all possible spanning trees of the graph.

Example Illustrations:

Example 1:

Consider a graph with vertices {A, B, C, D} and edges with weights:

- (A, B, 1)
- (A, C, 4)
- (B, C, 3)
- (B, D, 2)
- (C, D, 5)

1. **Initialization:**

- Sort edges: (A, B, 1), (B, D, 2), (B, C, 3), (A, C, 4), (C, D, 5)

Edge Selection:
- Add (A, B, 1) to the MST.
- Add (B, D, 2) to the MST.
- Add (B, C, 3) to the MST.
- Skip (A, C, 4) as it forms a cycle (A -> B -> C -> A).
- Skip (C, D, 5) as it forms a cycle (B -> C -> D -> B).

Resulting MST:
- MST edges: (A, B, 1), (B, D, 2), (B, C, 3)

Example 2:

Consider another graph with vertices {1, 2, 3, 4} and edges:

- (1, 2, 4)
- (1, 3, 1)
- (1, 4, 3)
- (2, 3, 2)
- (3, 4, 5)

1. **Initialization:**

- Sort edges: (1, 3, 1), (2, 3, 2), (1, 4, 3), (1, 2, 4), (3, 4, 5)

Edge Selection:
- Add (1, 3, 1) to the MST.
- Add (2, 3, 2) to the MST.
- Add (1, 4, 3) to the MST.
- Skip (1, 2, 4) as it forms a cycle (1 -> 3 -> 2 -> 1).
- Skip (3, 4, 5) as it forms a cycle (1 -> 3 -> 4 -> 1).

Resulting MST:
- MST edges: (1, 3, 1), (2, 3, 2), (1, 4, 3)

Conclusion:

Kruskal's Algorithm efficiently finds the Minimum Spanning Tree by iteratively selecting the smallest available edge that does not cause a cycle, thereby ensuring optimality. Its simplicity and effectiveness make it widely applicable in various fields such as network design, circuit routing, and clustering analysis.

References:

Shaffer, C. A. (2013). *A Practical Introduction to Data Structures and Algorithm Analysis* (3rd ed.). Dover Publications.

Wikipedia contributors. (2023). Kruskal's algorithm. In *Wikipedia, The Free Encyclopedia*. Retrieved from http://en.wikipedia.org/wiki/Kruskal's_algorithm

547 words

Permalink Show parent

Re: Week 4

by Natalie Tyson - Thursday, 18 July 2024, 4:37 AM

We are to describe Kruskal’s Minimum Spanning Tree algorithm for our discussion this week. This MST, or minimum spanning tree, has a weight that is less or equal to the weight of all of the other spanning trees. We sort the edges of the graph in increasing order and if the newly added edges do not form a cycle then we will add new edges and nodes. First we pick the edge that has the least weight to begin with and the last weighted edge picked is the one with the most weight. Since the algorithm will consistently make the optimal choice in every step to find the best solution it is called a Greedy algorithm.

How to use the algorithm to find the MST?

- Sort the edges from least to heaviest weighted edge
- Use the smallest edge to check if it has formed a cycle, if the cycle formed, we discard it, if it did not form, then the edge is included
- We will repeat the second step until there are only (v-1) edges left in the spanning tree, or we have touched on all the vertices

For the pseudo code we need to make sure that we are checking if adding the new edge will create a cycle or if it won't, we will always reject the edges that create the cycle.

We will look at some of the code formatting for Kruskal’s algorithm and focus on how to incorporate an MST into our programs.

Pseudo code for Kruskal format:

The following code allows us to incorporate another algorithm called Union Find, we had some of this brought up in our reading for class. This algorithm would divide the vertices into the clusters so that we could divide up the vertices more closely to see if they created a cycle or should get rejected. We have a small sample of this code below.

```
KRUSKAL (G):
B =  theta
For each vertex v G.V:
MAKE-SET(v)
For each edge (u, v) G.E ordered by increasing order by weight (u, v):
If FIND-SET FIND-SET (v):
A = A U {(u,v)}
UNION (u, v)
Return B
```

For Kruskal’s algorithm in Java we would create a graph by typing:

```
//Create a graph
Graph ( int v, int e) {
```

```
Vertices = v;

Edges = e;

Edge = new Edge[edges]

For (int i = 0; i < e; ++i)

Edge [ i ] = new Edge ();
```

Each vertices and edge would have to be entered into the graph like so:

```
Int vertices = 3; // Number of vertices

Int edges = 3; // Number of edges

Graph G = new Graph(vertices, edges);

G.edge [0].src = 0

G.edge [0]. Dest = 1;

G.edge [0].weight = 4

G.edge [1].src = 0

G.edge [1].dest = 2;

G.edge [1].weight = 4;

G.edge [2].src = 1;

G.edge [2].dest = 2;

G.edge [2].weight = 2;
```

We will also want to sort the edges

```
// Sorting edges

Arrays.sort(edge);

Subset subsets [ ] = new subset [vertices];

For (i = 0; i < vertices; ++i)

Subsets [ i ] = new subset ();

For (int v = 0; v < vertices; ++v) {

Subsets [v].parent = v;

Subsets [ v ].rank = 0;

}
```

The time complexity for Kruskal's Algorithm is O(E log E). We might use and apply this algorithm if we were laying electrical wiring or if you work in computer networking, it might be useful for certain applications.

Citations

- Kruskal's algorithm. Programiz. (n.d.). https://www.programiz.com/dsa/kruskal-algorithm
- Kruskal's minimum spanning tree (MST) algorithm. GeeksforGeeks. (2023, October 5). https://www.geeksforgeeks.org/kruskals-minimum-spanning-tree-algorithm-greedy-algo-2/#

587 words

Permalink Show parent

Re: Week 4
by Jobert Cadiz - Thursday, 18 July 2024, 5:11 AM

Hi Natalie,
Your response gives a good explanation of Kruskal's algorithm, including its greedy nature, how to build it, and useful applications. The quality of the response might be improved with a few little clarification changes, especially in the pseudo code. All things considered, it does a good job of conveying the main ideas of Kruskal's algorithm as well as its importance in computer science. Good work.

67 words

Permalink Show parent

Re: Week 4
by Prince Ansah Owusu - Thursday, 18 July 2024, 6:30 AM

our submission is comprehensive and well-structured. It effectively covers the essentials of Kruskal's algorithm and its implementation. Great job!

19 words

Permalink Show parent

Re: Week 4
by Liliana Blanco - Thursday, 18 July 2024, 9:20 AM

I appreciate your clear explanation of Kruskal's Algorithm. I like how you emphasized the importance of sorting edges and checking for cycles, which matches up with how I described the process. The pseudocode and Java examples you included give a practical perspective and show exactly how to use Kruskal's Algorithm. I also talked about how Kruskal's approach is kind of greedy, where it makes the best local choice by picking the shortest edge each time. Great work!

77 words

Permalink Show parent

Re: Week 4
by Jobert Cadiz - Thursday, 18 July 2024, 4:54 AM

Discussion Forum Unit 4

Kruskal's Algorithm: A Greedy Approach to Minimum Spanning Trees

Kruskal's algorithm is a popular method used to find the Minimum Spanning Tree (MST) of a connected, undirected graph. A collection of edges with the smallest possible total edge weight that joins every vertex to every other vertex is called the MST. Because Kruskal's method always selects the next shortest edge that does not create a cycle to build the MST, it is categorized as a greedy algorithm.

How Kruskal’s Algorithm Works

1. **Sort All Edges**: The first step in Kruskal’s algorithm is to sort all the edges in the graph in non-decreasing order of their weights.

2. **Initialize Subsets**: Create a subset for each vertex in the graph. This is often managed using a disjoint-set data structure (also known as union-find), which helps efficiently manage the merging of subsets and checking for cycles.

3. **Build the MST**: Iterate through the sorted edges and for each edge, check if the two vertices it connects are in different subsets:

o If they are in different subsets, add this edge to the MST and merge the two subsets.

o If they are in the same subset, adding this edge would form a cycle, so it is skipped.

4. **Repeat**: Continue this process until there are exactly V−1V - 1V−1 edges in the MST, where VVV is the number of vertices in the graph.

Why Kruskal’s Algorithm is ‘Greedy’

Kruskal’s algorithm is considered greedy because at each step it makes the locally optimal choice of selecting the smallest edge. This choice is made without considering the overall structure of the graph but only focusing on the immediate benefit of adding the smallest edge to the MST. The greedy strategy ensures that the total weight of the MST is minimized.

Example 1: Simple Graph

Let us consider a graph with 4 vertices (A, B, C, D) and the following edges with weights:

· A-B: 1

· A-C: 3

· A-D: 4

· B-C: 2

· B-D: 5

· C-D: 6

Steps:

1. **Sort Edges**: A-B (1), B-C (2), A-C (3), A-D (4), B-D (5), C-D (6)

2. **Initialize Subsets**: Each vertex is its own subset: {A}, {B}, {C}, {D}.

3. **Iterate and Add Edges**:

o A-B: Add (A and B are in different subsets) → MST: {A, B}

o B-C: Add (B and C are in different subsets) → MST: {A, B, C}

o A-C: Skip (A and C are in the same subset)

o A-D: Add (A and D are in different subsets) → MST: {A, B, C, D}

o B-D: Skip (B and D are in the same subset)

o C-D: Skip (C and D are in the same subset)

The MST is formed with edges A-B, B-C, and A-D with a total weight of 7.

Example 2: Complex Graph

Now, let us consider a more complex graph with 5 vertices (A, B, C, D, E) and the following edges with weights:

- A-B: 2
- A-C: 3
- B-C: 1
- B-D: 4
- C-D: 5
- C-E: 6
- D-E: 7

Steps:

1. **Sort Edges**: B-C (1), A-B (2), A-C (3), B-D (4), C-D (5), C-E (6), D-E (7)
2. **Initialize Subsets**: Each vertex is its own subset: {A}, {B}, {C}, {D}, {E}.
3. **Iterate and Add Edges**:

o B-C: Add (B and C are in different subsets) → MST: {B, C}

o A-B: Add (A and B are in different subsets) → MST: {A, B, C}

o A-C: Skip (A and C are in the same subset)

o B-D: Add (B and D are in different subsets) → MST: {A, B, C, D}

o C-D: Skip (C and D are in the same subset)

o C-E: Add (C and E are in different subsets) → MST: {A, B, C, D, E}

o D-E: Skip (D and E are in the same subset)

The MST is formed with edges B-C, A-B, B-D, and C-E with a total weight of 13.

Conclusion

By selecting the shortest edge at each stage, Kruskal's method perfectly illustrates the greedy approach and guarantees that the MST is built with the least amount of total weight feasible. Because of its effectiveness and ease of use, the algorithm is frequently used to solve MST issues in a variety of domains.

References

Shaffer, C. A. (2013). *Data Structures and Algorithm Analysis in Java* (3rd ed.). Dover Publications.

732 words

Permalink Show parent

Re: Week 4

by Prince Ansah Owusu - Thursday, 18 July 2024, 6:29 AM

Your explanation of Kruskal's Algorithm is thorough and well-structured, detailing each step clearly. The examples provided effectively illustrate how the algorithm selects edges to build the Minimum Spanning Tree (MST), demonstrating its greedy nature by always choosing the smallest available edge without creating cycles. The use of the disjoint-set data structure adds clarity to how subsets are managed to avoid cycles. Overall, it's a comprehensive explanation suitable for understanding Kruskal's Algorithm in depth. Well done!

75 words

Permalink Show parent

Re: Week 4

by Liliana Blanco - Thursday, 18 July 2024, 9:19 AM

Your breakdown of Kruskal’s Algorithm is really good. I like how you explained each step clearly and backed it up with examples. Your examples really help to show how the algorithm avoids cycles and builds the MST. Overall, your post gives a great analysis and emphasizes the strengths of Kruskal’s method in solving MST problems. Great job!

57 words

Permalink Show parent

Re: Week 4

by Tamaneenah Kazeem - Thursday, 18 July 2024, 10:54 AM

Excellent work

2 words

Permalink Show parent

Re: Week 4

by Liliana Blanco - Thursday, 18 July 2024, 5:58 AM

Kruskal's Algorithm finds the minimum spanning tree (MST) of a connected graph by adding edges one at a time to a growing forest, as long as adding a new edge does not create a cycle. It works by sorting all the edges and always adding the next lightest edge to the forest. This process continues until all the edges have been added, resulting in the MST. For example, if you have a graph with vertices A, B, C, and D, and edges with weights between them, Kruskal's Algorithm will start by adding the shortest edge, say between A and B. It then adds the next shortest edge that doesn't form a cycle, such as between B and C, until all vertices are connected without any cycles.

Greedy algorithms, as explained by Schaffer (2011), make the optimal local choice at each step to achieve a global optimum. They are used in various applications, including Kruskal's and Prim's algorithms for minimum spanning trees, Dijkstra's shortest paths, Huffman's coding, and different knapsack problems. They are also significant in addressing NP-completeness and optimization in computational problems.

Applying this to Kruskal's Algorithm, we see that it is a greedy algorithm because it always selects the shortest available edge to add to the MST without forming cycles. At each step, it makes the locally optimal choice by choosing the smallest edge, ensuring a global optimum, which is the MST.

References:

Schaffer, C. A. (2011). A practical introduction to data structures and algorithm analysis (3.1 ed.). Blacksburg, VA: Virginia Tech University, Department of Computer Science. Retrieved from http://people.cs.vt.edu/~shaffer/Book/C++3e20100119.pdf

261 words

Permalink Show parent

Re: Week 4

by Prince Ansah Owusu - Thursday, 18 July 2024, 6:25 AM

Your explanation of Kruskal's Algorithm effectively describes how it constructs a Minimum Spanning Tree by greedily selecting edges based on weight without forming cycles. Good use of examples and referencing Shaffer (2011) to explain the concept of greedy algorithms adds clarity. Well done!

43 words

Permalink Show parent

Re: Week 4

by Tamaneenah Kazeem - Thursday, 18 July 2024, 10:54 AM

Well done on this

4 words

Permalink Show parent

Re: Week 4

by Chong-Wei Chiu - Thursday, 18 July 2024, 8:11 AM

Kruskal's Algorithm:

This is a type of algorithm used to find the minimum spanning tree in a graph. Specifically, it is often employed for finding the minimum spanning tree in undirected graphs that may contain cycles. Such graphs typically represent complex problems with interconnected relationships. By applying this algorithm, we simplify the problem graph into its minimum spanning tree, focusing solely on the necessary connections.

Now, let's discuss how Kruskal's algorithm finds the minimum spanning tree from a graph:

1. List all the edges of the graph and sort them based on their weights.
2. Choose the edge with the smallest weight and include it in the spanning tree.
3. Proceed to the next smallest edge and check if adding it would create a cycle in the spanning tree formed so far. If adding the edge does not form a cycle, include it in the spanning tree. If it does form a cycle, skip this edge and move to the next one.
4. Repeat step 3 until all vertices are included in the spanning tree.

The definition of a minimum spanning tree is as follows:

1. It must include all vertices of the graph.
2. Its total weight (or cost) must be minimized.

To satisfy these conditions, we need at least $n-1$ edges (where n is the number of vertices), and these edges should have the smallest possible total weight. In step 1, listing all edges and sorting them by weight ensures that we can always select edges with the lowest weight available. Adding only one edge at a time ensures that each decision is optimal, which is the essence of a greedy algorithm. By continuing this process until all vertices are included, we ensure that the resulting tree has the minimum possible weight.

Reference:

Schaffer, C.A. (2011). A Practical Introduction to Data Structures and Algorithms Analysis (3.1 ed.). Blacksburg, VA: Virginia Tech University, Department of Computer Science. Available at http://people.cs.vt.edu/~shaffer/Book/C++3e20100119.pdf

Dasgupta, S., Papadimitriou, C.H., & Vazirani, U.V. (2006). Algorithms. Berkeley, CA: University of California Berkeley, Computer Science Division. Available at http://algorithmics.lsi.upc.edu/docs/Dasgupta-Papadimitriou-Vazirani.pdf

336 words

Permalink Show parent

Re: Week 4

by Tamaneenah Kazeem - Thursday, 18 July 2024, 10:26 AM

Describe in your own words how Kruskal’s algorithm works and be sure to articulate in your description what makes Kruskal’s algorithm ‘greedy’. Include one or two examples to explain your thought process to show what is occurring and how the methodology works.

Kruskal's algorithm is a simple greedy algorithm which is used to find the minimum spanning tree, also known as MST, of connected but undirected graphs.

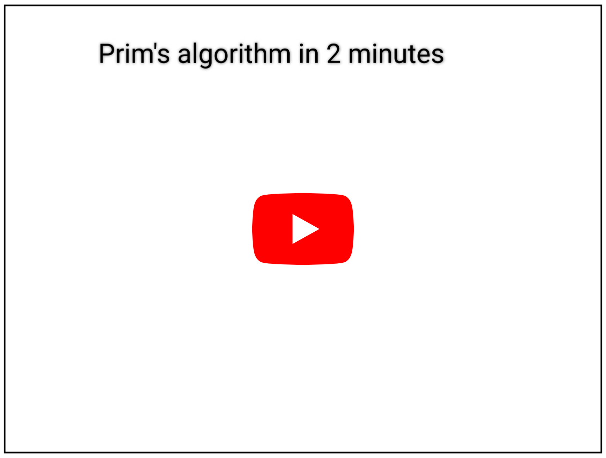
Here's how the Kruskal algorithm works:

1. Start with all edges of the graph and sort them in non-decreasing order by weight.
2. Identify the smallest edge and check if it forms a cycle with other edges in the MST. Add this edge if there is no cycle. Otherwise, leave it.
3. Keep repeating step 2
4. The algorithm will terminate when we have added $V-1$edges to the MST, where V is the number of vertices in the graph.

The reason for the name greedy in this context means that at each step of the algorithm it makes a locally optimal choice the hope of finding a globally optimal solution. The way Kruskal shows this is at each step it chooses the smallest possible edge which does not form a cycle with the previously selected edges. Doing this is to reveal the minimum spanning tree for the entire graph.

References:

Sambol, M. (2012b, October 28). *Prim’s algorithm in 2 minutes*. YouTube.



Chapter 11 Graphs, Sections 11.1 – 11.3 in A Practical Introduction to Data Structures and Algorithm Analysis by Clifford A. Shaffer.

248 words

Permalink Show parent