

December 11, 2023

Learning with ChatGPT

Hasse Diagram Scripting

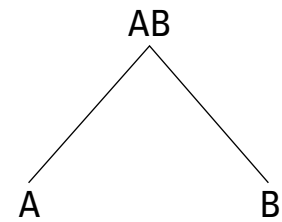
For an internal project, I was asked to visualize multiple 1-to-8-set Venn Diagrams, and display them in a poster.

One of the visualizations was in the form of a Hasse Diagram, which is just another way to visualize the traditional Venn Diagram.

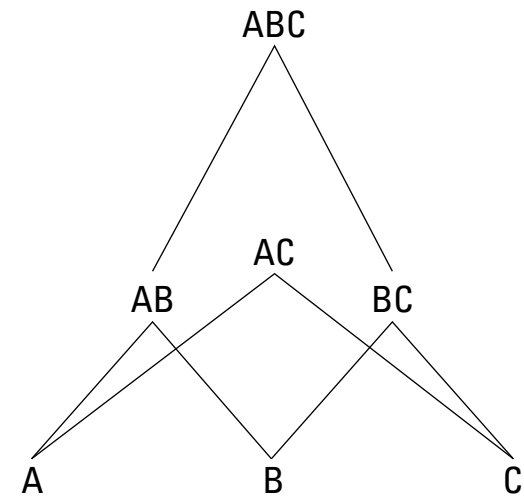
My boss started to create the diagram by hand —



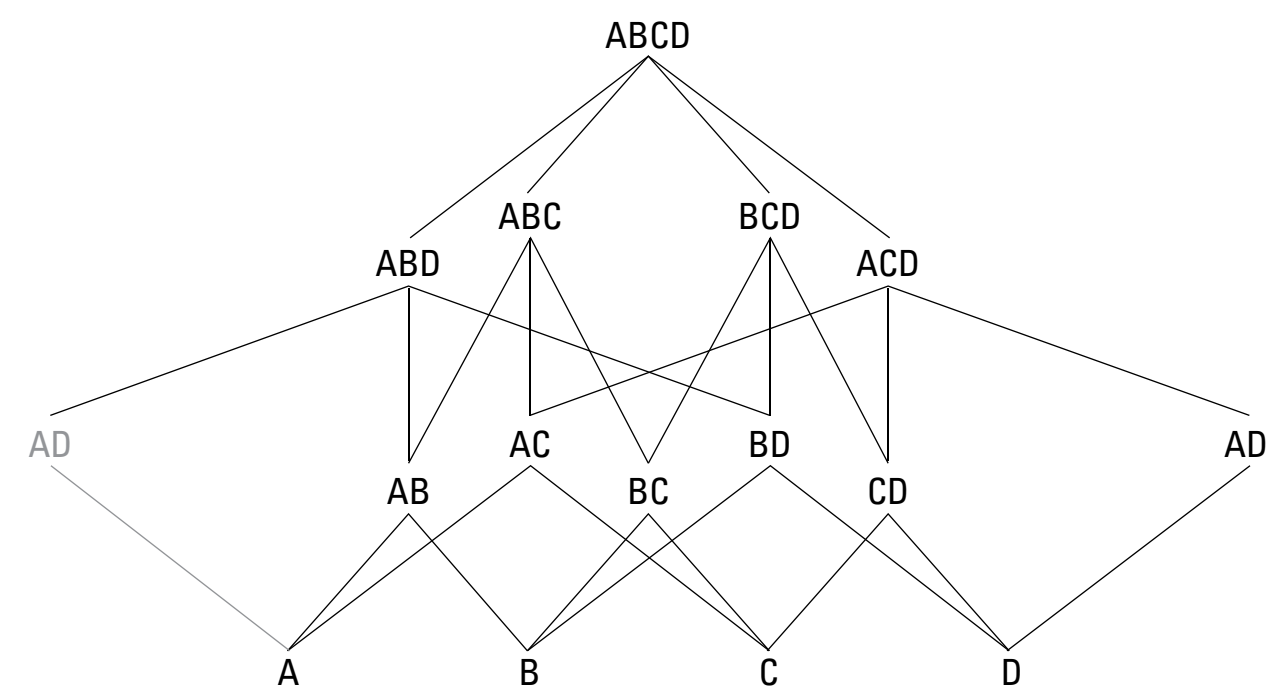
My boss started to create the diagram by hand — which grew,



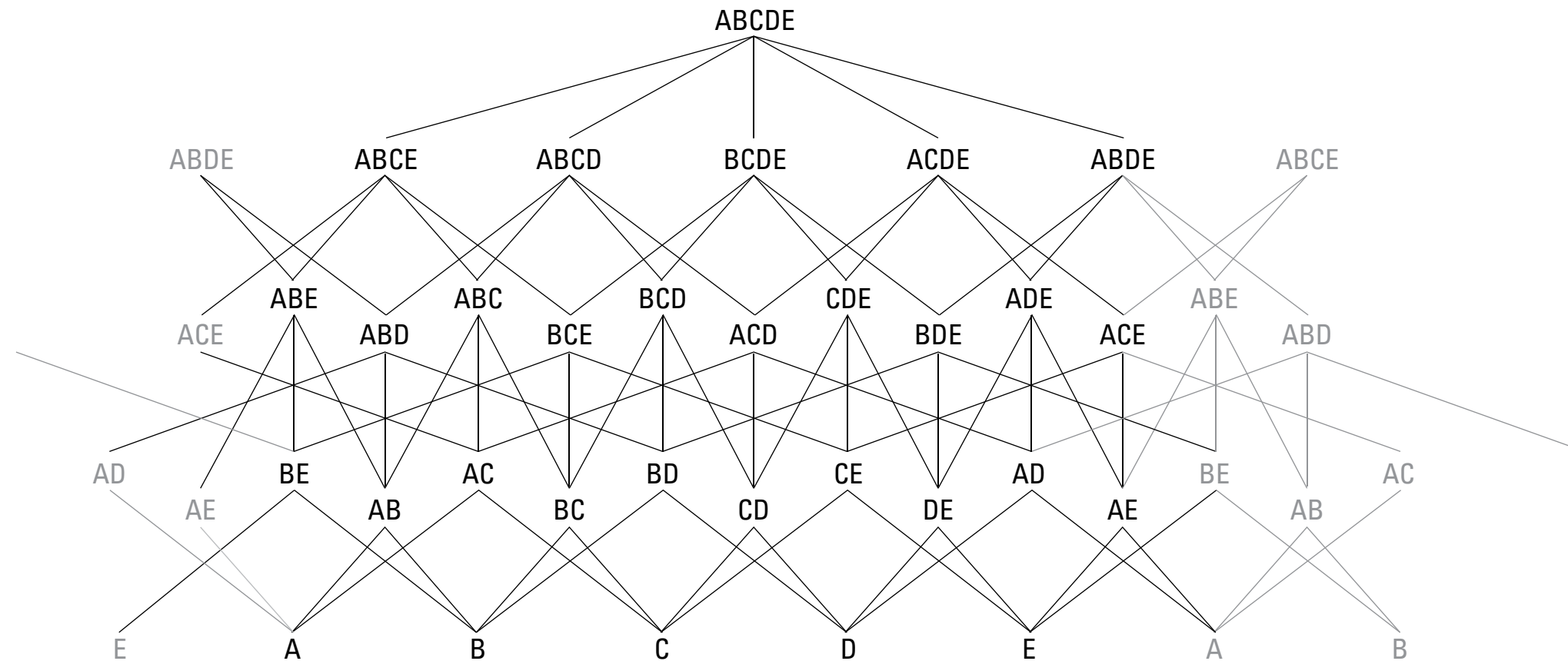
My boss started to create the diagram by hand — which grew, and grew,



My boss started to create the diagram by hand — which grew, and grew, and grew,



My boss started to create the diagram by hand — which grew, and grew, and grew, and grew some more.



To which he politely asked if I could finish where he started, to complete it out to 8-sets (he left off at 5-sets).

Unknowingly, this opened a portal for the evil villian “The Tedious Task” to cross over into our realm of peace.

So it was up to me to defeat him.

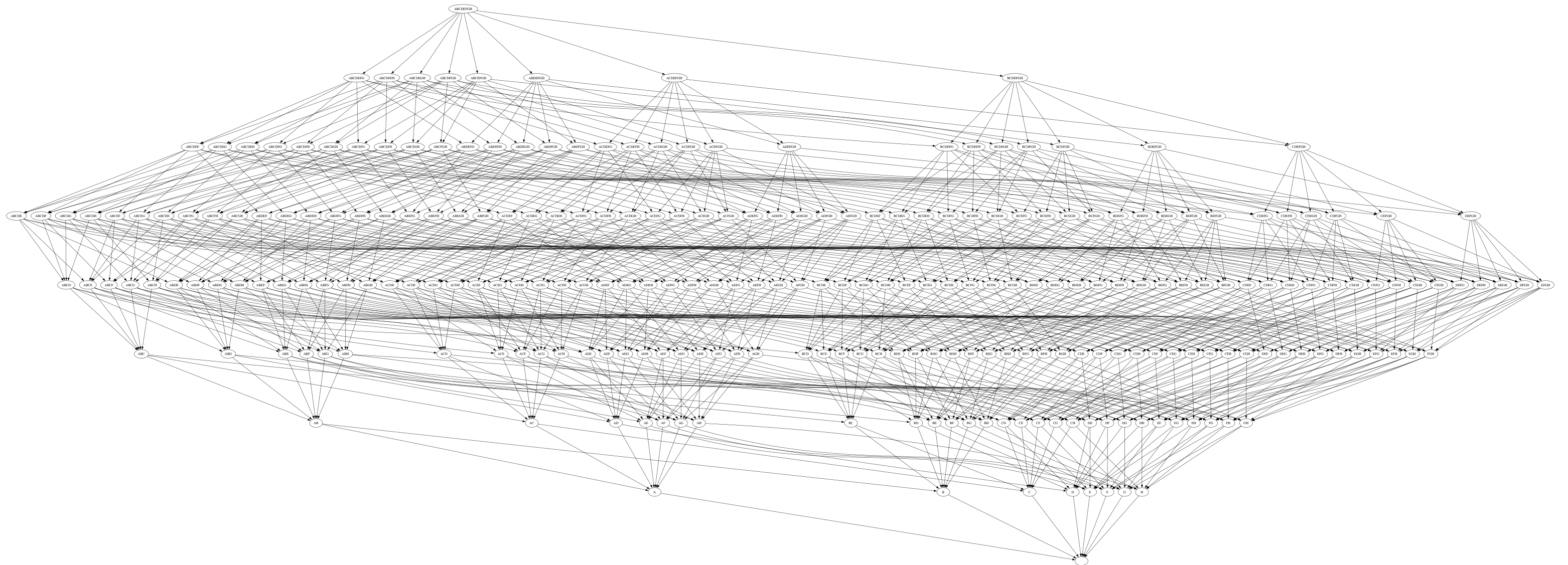
Piece of cake...



“The Tedious Task” never makes me waiver, as my sidekick is a perfect counter to him everytime he resurrects from his defeat.

My sidekick is very cryptic about their identity, so they go by... “ChatGPT”.

Not to my surprise, ChatGPT lent a very helpful hand, and together we created an almost flawless 8-set Hasse Diagram—defeating “The Tedious Task”... for now...



Appendix

Jokes aside, here is the process on how GPT helped me learn Python to create 1-8-set Hasse Diagrams.

With GPT's help, I was able to:

- Install Python
- Install Graphviz (enabling Python to render .dot files into viewable .png's)
- Run a script to render the .dot file
- Render a viewable .png of the Hasse Diagram I envisioned
- Render an editable .svg file
- Render .svg's for the prior 1-7-sets
- Fine tune small visual adjustments like spacing between rows and row orientation.

A few bonus steps it helped walk me through (though it might not have been needed as I look back):

- Installed the 'pip' package manager into my Python environment
- Backtracked due to a scary error message that there could be a conflict of package managers between 'pip' and my OS package manager — so it helped me setup a Virtual Environment to install 'pip' there instead.
- Installed the 'graphviz' package into my Virtual Environment from there.

Additional things I learned— whenever I would hit a road block with GPT, I would try referencing the bits that I learned from our conversation, or outright quoting / re-pasting the code it gave me.

For example, some of my prompts read:

“Thanks, so now where do I paste this in relation to the ‘hasse_diagram.dot’ code I chatted to you a few sections up our convo?”

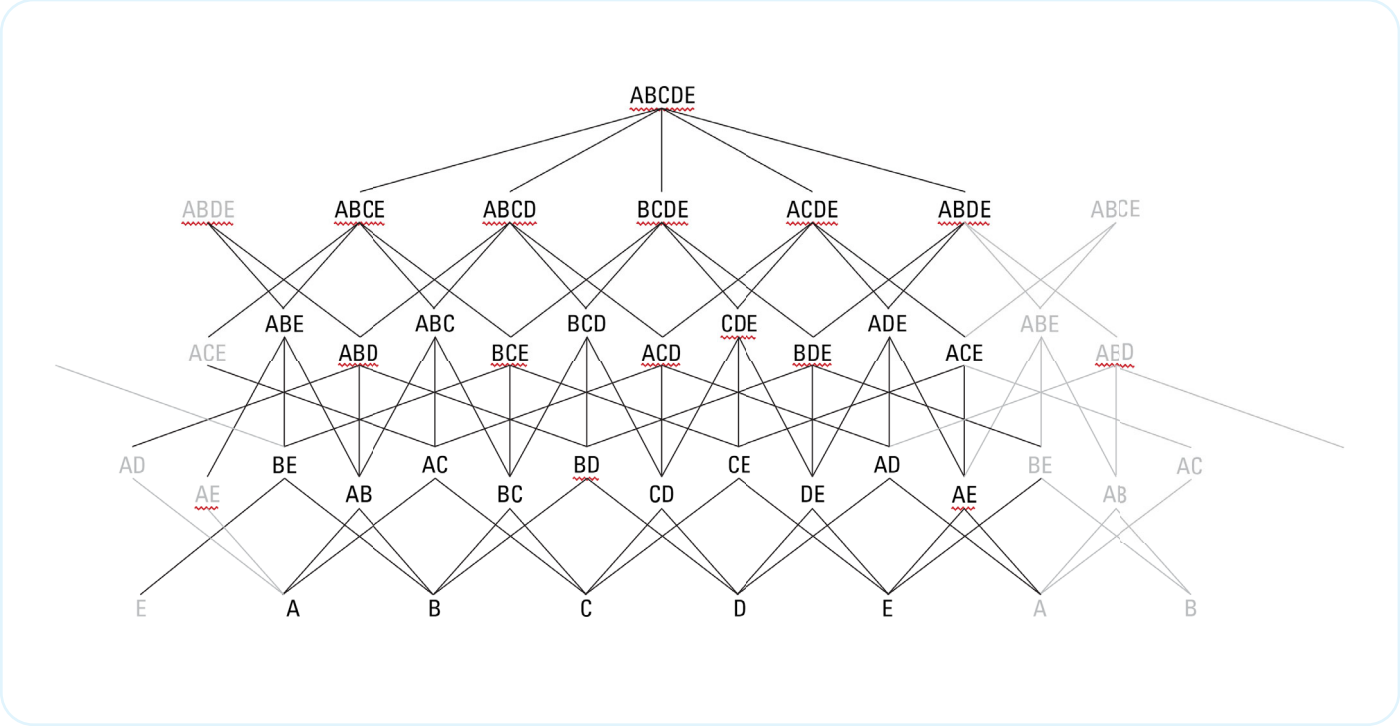
“Okay, so that didn’t quite work. How about this — take this code below that we worked together to create a few chats above, and flip the ‘rankdir’ direction. Here’s the code...” (pasted code from either our exchanges, or from my terminal)

To begin — with GPT-4, the option to upload images was now available.



So I began the conversation by uploading a screenshot of the 5-set Hasse diagram that I started with, and included a prompt that read:

Me:



The bottom row of this graph contains the letters A,B,C,D,E in black — extending upwards from those letter are the combinations of them. **Recreate a new graph, while adding the letters F, G, H, to the bottom row of the graph, extending upwards to show the newly added combinations.**

GPT said it can't create diagrams, but it offered to guide me on how to get the results I wanted. It recommends I try to programmatically create my diagram using tools that are meant for diagramming like Graphviz.



The bottom row of this graph contains the letters A,B,C,D,E in black— extending upwards from those letter are the combinations of them. **Recreate a new graph, while adding the letters F,G,H, to the bottom row of the graph, extending upwards to show the newly added combinations.**

GPT:

I cannot directly generate images. However, I can help guide you through the process of extending the graph:

The graph you provided showcases the Hasse diagram of the power set of the set {A, B, C, D, E}, minus the empty set. A Hasse diagram represents the partial order relation on a set.

To extend the graph with the letters F, G, and H:

Bottom Row:

Add F, G, H next to E. This will make the bottom row have the letters A, B, C, D, E, F, G, and H.

Second Row:

From each of F, G, and H, create upward connecting lines to the 2-combinations that include them, i.e., AF, BF, CF, DF, EF, AG, BG, etc.

Third Row:

Add 3-combinations that include at least one of the new letters F, G, or H. Examples are AFG, ABF, BFG, etc.

Continuing Upwards:

Similarly, you'd continue upwards, adding 4-combinations, 5-combinations, and so on, that include F, G, and H in various combinations with A, B, C, D, and E.

Top-most Row:

The top of your graph would have the set {A, B, C, D, E, F, G, H}, which represents the entire set. When drawing the connections (lines), ensure that there is a line from a subset to another if and only if the latter can be obtained by adding a single element to the former. For example, you'd have a line from A to AB, but not directly from A to ABC.

For the best representation, using graph visualization software or tools might be beneficial. Tools like Graphviz can help in generating such Hasse diagrams programmatically.

It didn't say Graphviz was free, so I asked it which free diagramming software does it recommend.

Bottom Row:

Add F, G, H next to E. This will make the bottom row have the letters A, B, C, D, E, F, G, and H.

Second Row:

From each of F, G, and H, create upward connecting lines to the 2-combinations that include them, i.e., AF, BF, CF, DF, EF, AG, BG, etc.

Third Row:

Add 3-combinations that include at least one of the new letters F, G, or H. Examples are AFG, ABF, BFG, etc.

Continuing Upwards:

Similarly, you'd continue upwards, adding 4-combinations, 5-combinations, and so on, that include F, G, and H in various combinations with A, B, C, D, and E.

Top-most Row:

The top of your graph would have the set {A, B, C, D, E, F, G, H}, which represents the entire set. When drawing the connections (lines), ensure that there is a line from a subset to another if and only if the latter can be obtained by adding a single element to the former. For example, you'd have a line from A to AB, but not directly from A to ABC.

For the best representation, using graph visualization software or tools might be beneficial. Tools like Graphviz can help in generating such Hasse diagrams programmatically.

Me:

Which software tools will create this diagram for me in a similar fashion of the screenshot I provided...**that are free to use.**

To which it listed some recommendations — Graphviz being it's top recommendation of the list again. So it is free.

For the best representation, using graph visualization software or tools might be beneficial. Tools like Graphviz can help in generating such Hasse diagrams programmatically.

GPT:

If you're looking for free software tools to create such a Hasse diagram, here are some recommendations:

Graphviz:

Graphviz is a collection of graph drawing tools and libraries. It uses the "dot" language to define graphs, and you can render those definitions into graphical images. You would need to manually specify the nodes and edges based on your sets, but once you've got the hang of it, it's a powerful tool.

SageMath:

SageMath is a free, open-source mathematics software system that integrates several mathematics libraries into a single experience. It can be used to work with combinatorics and can create Hasse diagrams for posets. Using SageMath, you can define a poset based on the subsets of your set and then visualize it as a Hasse diagram.

Draw.io (now diagrams.net):

While not specialized for Hasse diagrams, this is a free, online diagramming tool that can be used for all sorts of flowcharts, diagrams, etc. You would need to manually draw the nodes and connections, but it provides a lot of flexibility in layout and appearance.

LaTeX with TikZ or pgfplots package:

If you're familiar with LaTeX, the TikZ or pgfplots packages can be used to draw intricate diagrams, including Hasse diagrams. There's a learning curve, but it gives you a lot of control over the appearance and layout of the diagram.

For the easiest user experience without coding, diagrams.net might be the best choice. However, if you're looking for automation or have a mathematical bent, Graphviz or SageMath could be more suitable. If you're already familiar with LaTeX, using TikZ or pgfplots is a powerful option.

Me:

Which software tools will create this diagram for me in a similar fashion of the screenshot I provided...that are free to use.

Still on the assumption that there would be a lot of manual work, I ask which of the offerings work with code, and if it can write it for me.

those definitions into graphical images. You would need to manually specify nodes and edges based on your sets, but once you've got the hang of it, it's a powerful tool.

SageMath:

SageMath is a free, open-source mathematics software system that integrates several mathematics libraries into a single experience. It can be used to work with combinatorics and can create Hasse diagrams for posets.

Using SageMath, you can define a poset based on the subsets of your set and then visualize it as a Hasse diagram.

Draw.io (now diagrams.net):

While not specialized for Hasse diagrams, this is a free, online diagramming tool that can be used for all sorts of flowcharts, diagrams, etc. You would need to manually draw the nodes and connections, but it provides a lot of flexibility in layout and appearance.

LaTeX with TikZ or pgfplots package:

If you're familiar with LaTeX, the TikZ or pgfplots packages can be used to draw intricate diagrams, including Hasse diagrams. There's a learning curve, but it gives you a lot of control over the appearance and layout of the diagram.

For the easiest user experience without coding, diagrams.net might be the best choice. However, if you're looking for automation or have a mathematical bent, Graphviz or SageMath could be more suitable. If you're already familiar with LaTeX, using TikZ or pgfplots is a powerful option.

Me:

Okay, first, **which of these offerings enables me to paste in a code block** to draw this diagram.

Second, **could you write the code to create such a diagram** to paste into one of the software offerings you mention above?

To which it only writes the actual code partially— though it follows up with an offer to write a Python script, that would then generate the code I needed.

GPT:

Of the tools I mentioned:

- 1. Graphviz uses the “dot” language for defining graphs.
- 2. SageMath allows for mathematical and Pythonic definitions of posets and their Hasse diagrams.
- 3. LaTeX with TikZ or pgfplots uses the LaTeX language and the respective packages’ commands.

Code for the Diagram:

For simplicity and ease of understanding, I’ll provide an example using Graphviz. Here’s a basic structure for the original diagram (ABCDE):

```
dot
Copy code
digraph G {
  // Define nodes
  "ABCDE" -> {"ABCD" "BCDE" "ACDE" "ABDE" "ABCE"}
  "ABCD" -> {"ABC" "ABD" "ACD" "BCD"}
  "BCDE" -> {"BCD" "BCE" "CDE" "BDE"}
  // ... continue this for all combinations down to single letters

  // Styling
  rankdir=BT; // Bottom to top
}
```

You’d continue specifying the nodes and their connections in this manner. Graphviz will handle the layout for you.

For extending the diagram to include F, G, and H, you’d similarly add nodes and connections for those combinations.

However, it’s worth noting that the total number of combinations will grow substantially as you add more letters. Writing this out by hand will be tedious, so automating it (e.g., with a Python script) might be more efficient.

Unfortunately, due to the combinatorial nature of the task, providing the complete code for the diagram with A through H here would be lengthy and complex. If you’re interested in an automated solution or a different tool, please let me know!

Me:

Okay, first, which of these offerings enables me to paste in a code block to draw this diagram.

Second, could you write the code to create such a diagram to paste into one of the software offerings you mention above?

I try to rephrase my prompt by specifically asking it to write the “.dot” code now that I knew the proper term of art. Though I also pose the automation route as option 2.

You'd continue specifying the nodes and their connections in this manner. Graphviz will handle the layout for you.

For extending the diagram to include F, G, and H, you'd similarly add nodes and connections for those combinations.

However, it's worth noting that the total number of combinations will grow substantially as you add more letters. Writing this out by hand will be tedious, so automating it (e.g., with a Python script) might be more efficient.

Unfortunately, due to the combinatorial nature of the task, providing the complete code for the diagram with A through H here would be lengthy and complex. If you're interested in an automated solution or a different tool, please let me know!

Me:

First, can you rewrite the dot code above to contain A, B, C, D, E, F, G, H, so I can then past it into Graphviz.

Second, if that doesn't work, what are some alternate options for an automated solution to create the diagram from A through H, so I don't have to do it by hand.

Again... it won't write the .dot code— but I have a Python script now to do so.

GPT:

Writing the dot code for all the combinations of A through H is a combinatorially complex task. However, I can provide you with a basic structure to get started and automate the process using Python to generate the dot file.

Dot Structure for A through H (partial):

```
dot
Copy code
digraph G {
    // Define nodes
    "ABCDEFGH" -> {"ABCDEFGH" "BCDEFGH" "ACDEFGH" "ABDEFGH" ...}
    "ABCDEFG" -> {"ABCDEF" "BCDEFG" "ACDEFG" ...}
    // ... This pattern will be very lengthy and continues for all combinations down to single letters

    // Styling
    rankdir=BT; // Bottom to top
}
```

The above is just a structure, and filling out the “...” portions manually would be a massive task.

Automated Solution Using Python:

Here's a Python script that generates the dot code for combinations from A through H:

```
python
Copy code
from itertools import combinations

letters = "ABCDEFGH"
dot_code = "digraph G {\n"
dot_code += '    rankdir=BT; // Bottom to top\n'

for i in range(len(letters), 0, -1):
    for combo in combinations(letters, i):
        parent = ''.join(combo)
        children = [''.join(c) for c in combinations(parent, i-1)]
        if children:
            dot_code += '    "{}" -> {{{}}}\n'.format(parent, ''.join(['{}'.format(child) for child in children]))

dot_code += "}\n"

with open('hasse_diagram.dot', 'w') as f:
    f.write(dot_code)
```

Run the above script, and it will generate a file named `hasse_diagram.dot`. You can then input this file into Graphviz to produce the diagram.

If you prefer another automated solution, a Python script using `networkx` and `matplotlib` could be used to directly visualize the graph, but the output might not be as clean and organized as with Graphviz.

Seemed, if that doesn't work, what I'll do is write a script to generate the diagram for A through H, so I don't have to do it by hand.

The problem is I have zero knowledge of Python, hence why I was adamant about not taking the automation route. But on we go.

```
for i in range(len(letters), 0, -1):
    for combo in combinations(letters, i):
        parent = "".join(combo)
        children = ["".join(c) for c in combinations(parent, i-1)]
        if children:
            dot_code += ' "{}" -> {{{}}}\n'.format(parent, ' '.join(["{}".format(child) for child in children]))

dot_code += "}\n"

with open('hasse_diagram.dot', 'w') as f:
    f.write(dot_code)
```

Run the above script, and it will generate a file named `hasse_diagram.dot`. You can then input this file into Graphviz to produce the diagram.

If you prefer another automated solution, a Python script using `networkx` and `matplotlib` could be used to directly visualize the graph, but the output might not be as clean and organized as with Graphviz.

Me:

Thanks. Though I've never used Python, where should I start? Is it a download or what?

To my surprise, it exceeded my expectations, and gave me super clear instructions on getting started with Python.

If you prefer another automated solution, a Python script using `networkx` and `matplotlib` could be used to directly visualize the graph, but the output might not be as clean and organized as with Graphviz.

GPT:

Absolutely! Python is one of the most versatile and beginner-friendly programming languages. Here's a basic step-by-step guide to get you started:

Download and Install Python:

Go to the official Python website's download page: [Python Downloads](#)

Download the latest version for your operating system (Windows, macOS, Linux).

Run the installer. On Windows, ensure you check the box that says "Add Python to PATH" during installation. This will make it easier to run Python from the command line.

Using Python:

Windows: You can use the Python IDLE (which comes with the Python installation) or the command prompt (just type `python`).

macOS/Linux: Open your terminal and type `python3`.

Installing Necessary Libraries:

For our script, you don't need additional libraries, but in general, Python libraries can be installed using `pip`, which is the package installer for Python.

For example:

```
pip install library_name
```

If you later decide to use the `networkx` and `matplotlib` solution, you'd install them like (bash):

```
pip install networkx matplotlib
```

Running the Script:

Save the Python script I provided (or any other Python script) with a `.py` extension, for example: `generate_graph.py`.

Navigate to the directory containing your script using the command prompt or terminal.

Run the script by typing:

```
python generate_graph.py
```

If you're on macOS/Linux, you might need to use `python3` instead of `python`.

Learning Python:

If you're interested in learning Python, there are numerous online resources, courses, and tutorials available. Some popular ones include:

Python.org's Beginner Guide

Real Python

Automate the Boring Stuff with Python

Me:

Thanks. Though I've never used Python, where should I start? Is it a download or what?

But... there were some kinks that needed to be worked out. I didn't know how to save the script it gave me— again, zero knowledge.

Run the script by typing:

```
python generate_graph.py
```

If you're on macOS/Linux, you might need to use python3 instead of python.

Learning Python:

If you're interested in learning Python, there are numerous online resources, courses, and tutorials available. Some popular ones include:

Python.org's Beginner Guide

Real Python

Automate the Boring Stuff with Python

Advanced (Optional):

If you plan on working on multiple Python projects or with different versions of Python, consider using virtual environments or tools like Pyenv to manage multiple Python versions.

Once you've got Python set up and the script saved, you can run it, and it'll generate the `hasse_diagram.dot` file for you. After that, you can feed the `.dot` file into Graphviz to visualize the diagram.

Me:

How do I save the python script you provided?

So GPT walked me through that too.

Additionally, it restated how to run the script...

Advanced (Optional):

If you plan on working on multiple Python projects or with different versions of Python, consider using virtual environments or tools like Pyenv to manage multiple Python versions.

Once you've got Python set up and the script saved, you can run it, and it'll generate the `hasse_diagram.dot` file for you. After that, you can feed the `.dot` file into Graphviz to visualize the diagram.

GPT:

To save the Python script I provided:

Using a Text Editor:

- Open a text editor. This could be Notepad on Windows, TextEdit on macOS, or any other plain text editor.
- Copy the Python script from my previous response.
- Paste the script into the text editor.
- Save the file with a `.py` extension, for instance, `generate_graph.py`.

Location:

You can save it on your Desktop or any directory you prefer. Just remember the location because you'll need to navigate to it in the command prompt or terminal to run the script.

Running the Script:

Once saved, open your command prompt (Windows) or terminal (macOS/Linux).

Navigate to the directory where you saved the file. For instance, if you saved it on your Desktop:

On Windows: `cd Desktop`

On macOS/Linux: `cd ~/Desktop`

Run the script by typing:

`python generate_graph.py`

On macOS/Linux, you might use `python3` instead of `python`:

`python3 generate_graph.py`

After running the script, it will create a "`hasse_diagram.dot`" file in the same directory, which you can then input into Graphviz.

If you've never worked with text files or the command line before, this might feel a bit unfamiliar, but with practice, it becomes straightforward!

Me:

How do I save the python script you provided?

Which I was able to follow. So now I have some .dot code!

Running the Script:

Once saved, open your command prompt (Windows) or terminal (macOS/Linux).

Navigate to the directory where you saved the file. For instance, if you saved it on your Desktop:

On Windows: `cd Desktop`

On macOS/Linux: `cd ~/Desktop`

Run the script by typing:

`python generate_graph.py`

On macOS/Linux, you might use python3 instead of python:

`python3 generate_graph.py`

After running the script, it will create a “hasse_diagram.dot” file in the same directory, which you can then input into Graphviz.

If you’ve never worked with text files or the command line before, this might feel a bit unfamiliar, but with practice, it becomes straightforward!

Me:

Okay, now I have the hasse_diagram.dot on my desktop now, now what’s next to put it into Graphviz?

GPT then walked me through installing Graphviz, which is a diagramming software that Python uses to generate the diagrams I needed.

Graphviz.

If you've never worked with text files or the command line before, this might feel a bit unfamiliar, but with practice, it becomes straightforward!

GPT:

Great to hear you got it working! Here's how to visualize the `hasse_diagram.dot` file using Graphviz:

Install Graphviz:

If you haven't already installed Graphviz, you can download it from Graphviz's official website. Choose the appropriate version for your operating system and follow the installation instructions.

Using Graphviz with the Command Line:

Open your command prompt (Windows) or terminal (macOS/Linux).
Navigate to the directory where you saved the `hasse_diagram.dot` file.

If it's on your Desktop:
On Windows: `cd Desktop`
On macOS/Linux: `cd ~/Desktop`

Use the `dot` command (part of the Graphviz toolset) to convert the `.dot` file into an image format of your choice.

For instance, to create a PNG image:

```
dot -Tpng hasse_diagram.dot -o hasse_diagram.png
```

This command will generate a PNG image named `hasse_diagram.png` in the same directory.

Viewing the Diagram:

You can now open the generated `hasse_diagram.png` image with any image viewer to see the visualized Hasse diagram.

Alternative: Using Graphviz GUI Applications:

Some Graphviz installations come with GUI applications (like `gvedit` on Windows) that allow you to open `.dot` files and visualize them directly without using the command line. If you have one of these applications, you can simply open the `hasse_diagram.dot` file with it and view or export the visualization.

That's it! Once you've visualized the diagram, you can further customize or adjust the appearance as needed using Graphviz's extensive features and settings.

Me:

Okay, now I have the `hasse_diagram.dot` on my desktop now, now what's next to put it into Graphviz?

And just like that, I hit a milestone. I have an image of the Hasse Diagram I needed!

Use the dot command (part of the Graphviz tools) to convert the .dot file into an image format of your choice.

For instance, to create a PNG image:
`dot -Tpng hasse_diagram.dot -o hasse_diagram.png`

This command will generate a PNG image named hasse_diagram.png in the same directory.

Viewing the Diagram:
You can now open the generated hasse_diagram.png image with any image viewer to see the visualized Hasse diagram.

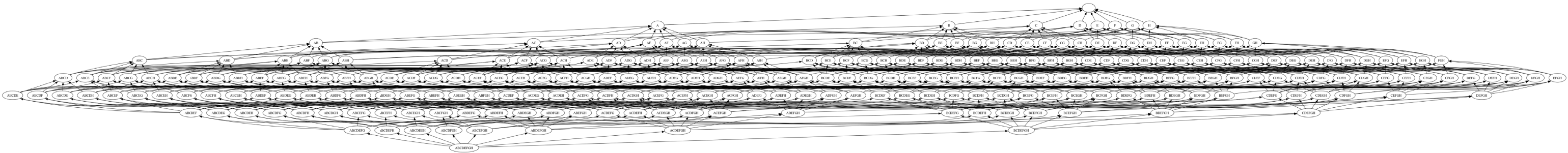
Alternative: Using Graphviz GUI Applications:
Some Graphviz installations come with GUI applications (like gvedit on Windows) that allow you to open .dot files and visualize them directly without using the command line. If you have one of these applications, you can simply open the hasse_diagram.dot file with it and view or export the visualization.

That's it! Once you've visualized the diagram, you can further customize or adjust the appearance as needed using Graphviz's extensive features and settings.

Me:

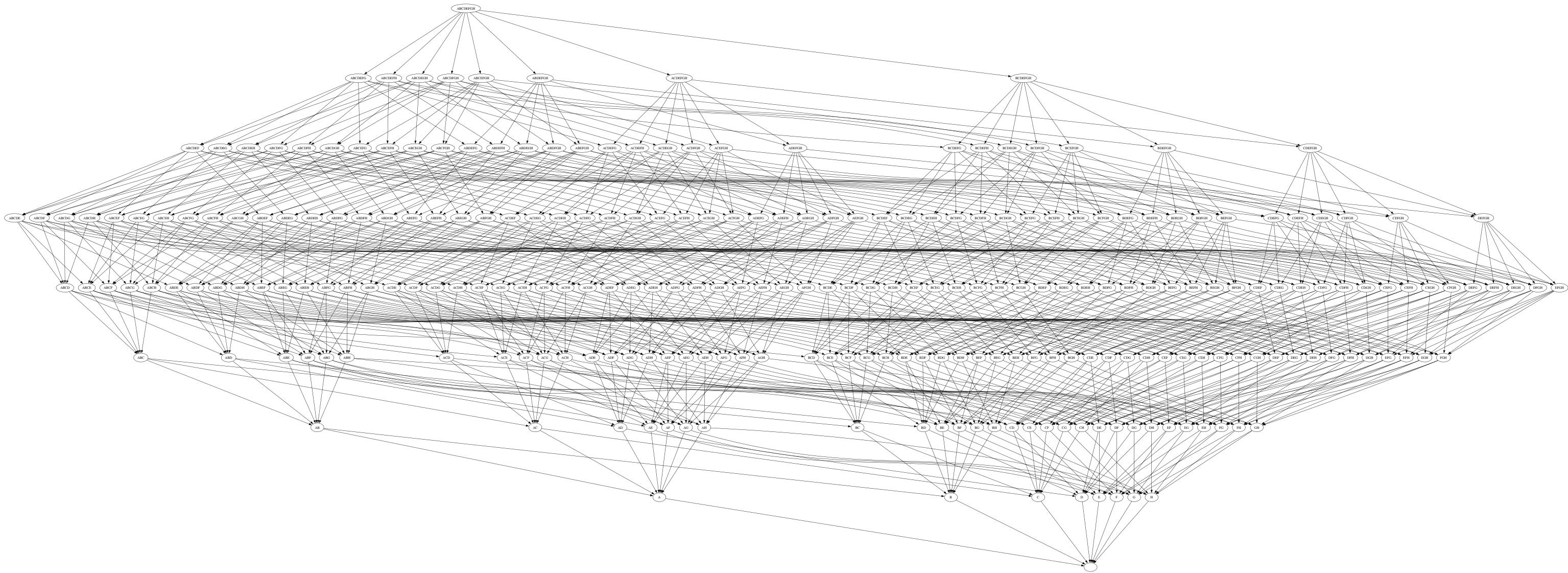
Your instructions worked to create a png of the diagram I was envisioning, thanks!

Though it could use some work...



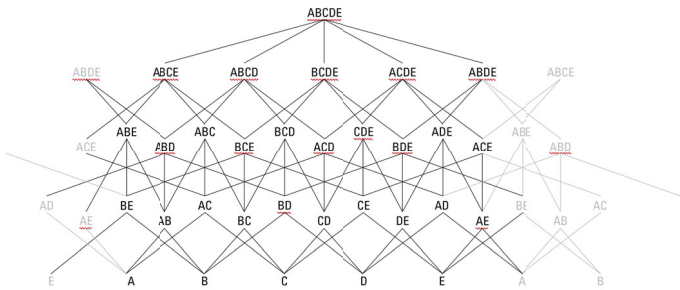
But I ultimately was able to fine tune my diagram.

‘Final’ Diagram
After a few rounds with GPT.



Quite the leap from the beginning.

Starting point
5-set Hasse Diagram (A, B, C, D, E) — Hand Drawn over a span of hours



Final State
8-set Hasse Diagram (A, B, C, D, E, F, G, H) — GPT x Python Drawn in seconds after setup.

