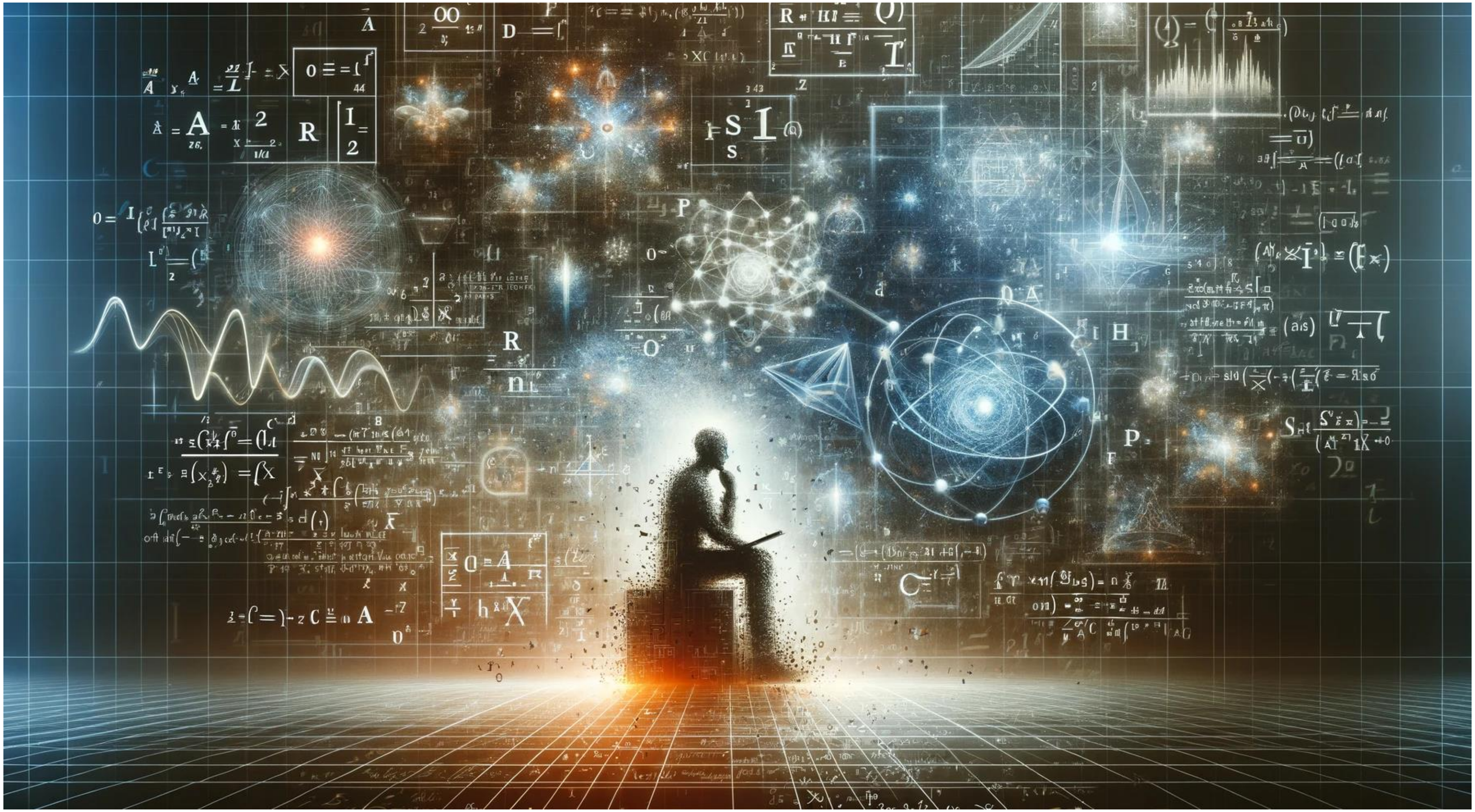
OMIS 6000

Week 2:

- The Python Programming Language
- Optimization with Python (Gurobi)
- Transportation and Transshipment
- Multi-period Optimization Models
- Large Language Models (LLMs)

Adam Diamant, Schulich School of Business, Winter 2025

Optimization in Python



Optimization in Python

Packages for Linear Models:

- PuLP

Packages for Nonlinear Models:

- pyOpt
- CVXOPT
- ipopt

Packages for both:

- Gurobi (Lots of documentation @ Knowledge Center)
- pyscipopt

Gurobi for Python

- A commercial optimization tool that is free for academic use and has Python bindings. It is the the fastest optimization solver in the world!

Installing Gurobi

- Register for a Gurobi account here using your Schulich email (schulich.yorku.ca):
  - Make sure to select **Academic** user.
- Download the Latest Version of Gurobi of the **Gurobi Optimizer** and install.
  - Make sure to select **Academic** user.
- Request an Academic License.
  - Open a terminal/command prompt window and run '*grbgetkey*' using the argument provided to register your machine.

Installing Gurobi

- This Gurobi academic license can be set up only a single physical machine. Users may install and license Gurobi for their own use on more than one machine.
- You will be able to access and use Gurobi as long as your have a Schulich email.
- Licenses must be renewed every year.
  - The academic licensing program gives you free access to commercial software that is used by more than 2,400 companies.

Installing Gurobi Python

The Guorbipy python library must be installed before using the software to communicate with the Gurobi optimization suite.

- If you have installed Anaconda, type the following into the Spyder console window.

```
conda install -c gurobi gurobi
```

- If you are using another IDE, type this into the Spyder console. Make sure you have pip.

```
pip install gurobipy
```

Installing Gurobi Python

The Guorbipy python library must be installed before using the software to communicate with the Gurobi optimization suite.

- If you have installed Anaconda, open an Anaconda Prompt window and type:

```
conda install -c gurobi gurobi
```

- If you are using another IDE, open a terminal/command prompt window, scroll to the corresponding Python directory and type:

```
python –m pip install gurobipy
```

Gurobi for Python

There are 3 components to a Python model:

A. The Gurobi model object represents the optimization model expressed in Python.

B. Elements of the optimization model.
   1. Decision Variables
   2. Objective Function
   3. Constraints

C. An optimized model and the methods that support inspecting the optimal solution.

Linear Programming with Gurobi and Python

LG Production Example

Transportation and Transshipment

Multi-Period Optimization

LG Production Example

LG Production Example

LG is one of the world's largest LCD panel maker, in particular, those based on LED's *(light-emitting diodes)*. The company recently received an order for three types of LED displays. Each display type requires time for assembling and packaging. The table summarizes the requirements for each LED type.

	Type A	Type B	Type C
Numbers Ordered	3,000	2,000	900
Assembling (Hours)	2	1.5	3
Packaging (Hours)	1	2	1
Profit per unit	$60	$75	$80

LG has only 10,000 hours of assembly and 5,000 hours of packaging available which is not enough to satisfy all orders.

How many units of each type should LG produce to maximize profits?

LG Production Example

Final Formulation

Maximize $Z =$

(Assembly)

(Packaging)

(Order limits)

$x_A \geq 0, \quad x_B \geq 0, \quad x_C \geq 0$ *(Non-negativity)*

Optimal Solution:

LG Production Example

Final Formulation

Maximize $Z = 60x_A + 75x_B + 80x_C$

(Assembly)

(Packaging)

(Order limits)

$x_A \geq 0, \quad x_B \geq 0, \quad x_C \geq 0$ *(Non-negativity)*

Optimal Solution:

LG Production Example

Final Formulation

Maximize $\boldsymbol{Z = 60x_A + 75x_B + 80x_C}$

$\boldsymbol{2x_A + 1.5x_B + 3x_C \leq 10,000}$ ***(Assembly)***

$\boldsymbol{x_A + 2x_B + x_C \leq 5,000}$ ***(Packaging)***

$\boldsymbol{x_A \leq 3000, x_B \leq 2000, x_C \leq 900}$ ***(Order limits)***

$\boldsymbol{x_A \geq 0, \quad x_B \geq 0, \quad x_C \geq 0}$ ***(Non-negativity)***

<u>Optimal Solution:</u>

LG Production Example

Final Formulation

Maximize $Z = 60x_A + 75x_B + 80x_C$

$2x_A + 1.5x_B + 3x_C \leq 10{,}000$ *(Assembly)*

$x_A + 2x_B + x_C \leq 5{,}000$ *(Packaging)*

$x_A \leq 3000, x_B \leq 2000, x_C \leq 900$ *(Order limits)*

$x_A \geq 0, \quad x_B \geq 0, \quad x_C \geq 0$ *(Non-negativity)*

Optimal Solution: *The highest profit that can be obtained is $293,250 producing 3000 units of type A, 550 units of type B, and 900 units of type C.*

LG Production Example: Python Solution

- Make sure you remember to actually solve the Gurobi model after formulating it.
- When/if an error occurs, pay attention to what line is generating the error.
  - Typically, errors are syntax-related.
  - You can also progressively add constraints (comment the others out) to the problem until you figure out which constraint is problematic.

What managerial intuition do you get from the Python solution?

How does Gurobi do it?

Basic concept of the Simplex algorithm:

1. Start at a vertex (corner point).
2. If there is an adjacent corner point that yields a better objective function value, move to that point.
3. Otherwise, stop. The current solution is optimal.

A quick description of the Simplex algorithm including the Simplex Tableau as well as basic and non-basic solutions can be found here.

How does Gurobi do it?

Basic concept of the Simplex algorithm:

1. Start at a vertex (corner point).
2. If there is an adjacent corner point that yields a better objective function value, move to that point.
3. Otherwise, stop. The current solution is optimal.

Theorem: If the Simplex algorithm does not contain cycles (i.e., there is no degeneracy), it will monotonically converge to the optimal solution in a finite number of iterations. No cycling can be guaranteed using Bland's Rule.

How does Gurobi do it?

Basic concept of the Simplex algorithm:

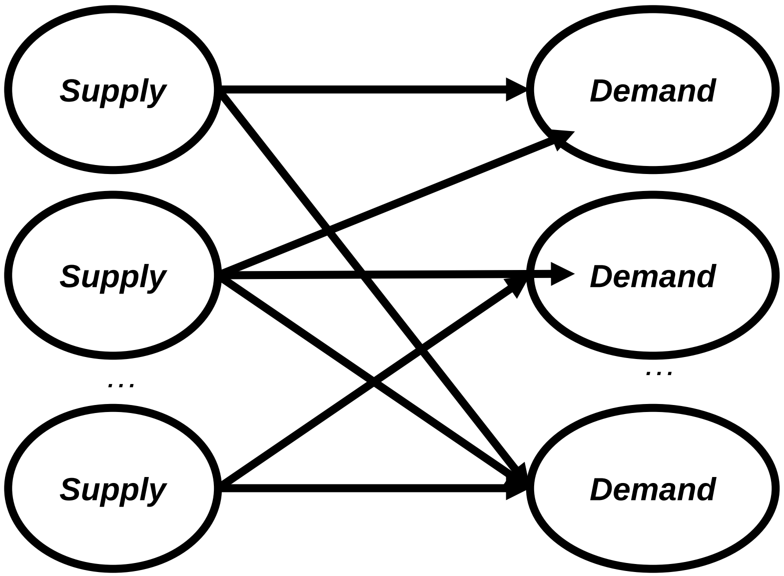
1. Start at a vertex (corner point).
2. If there is an adjacent corner point that yields a better objective function value, move to that point.
3. Otherwise, stop. The current solution is optimal.

Proof of Convergence: In-Class.

Transportation Example

Transportation Example

- Two classes of nodes: **supply and demand**



- The objective is to transport products/goods from the supply nodes to the demand nodes.

Transportation Example

The juice company ***Grovestand***® has three orange groves and three processing plants. Each grove has oranges that must enter the market immediately (storage would degrade the quality of the orange). Each plant has a processing capacity. The amount of supply, capacity, and the distances between the groves and the plants are:

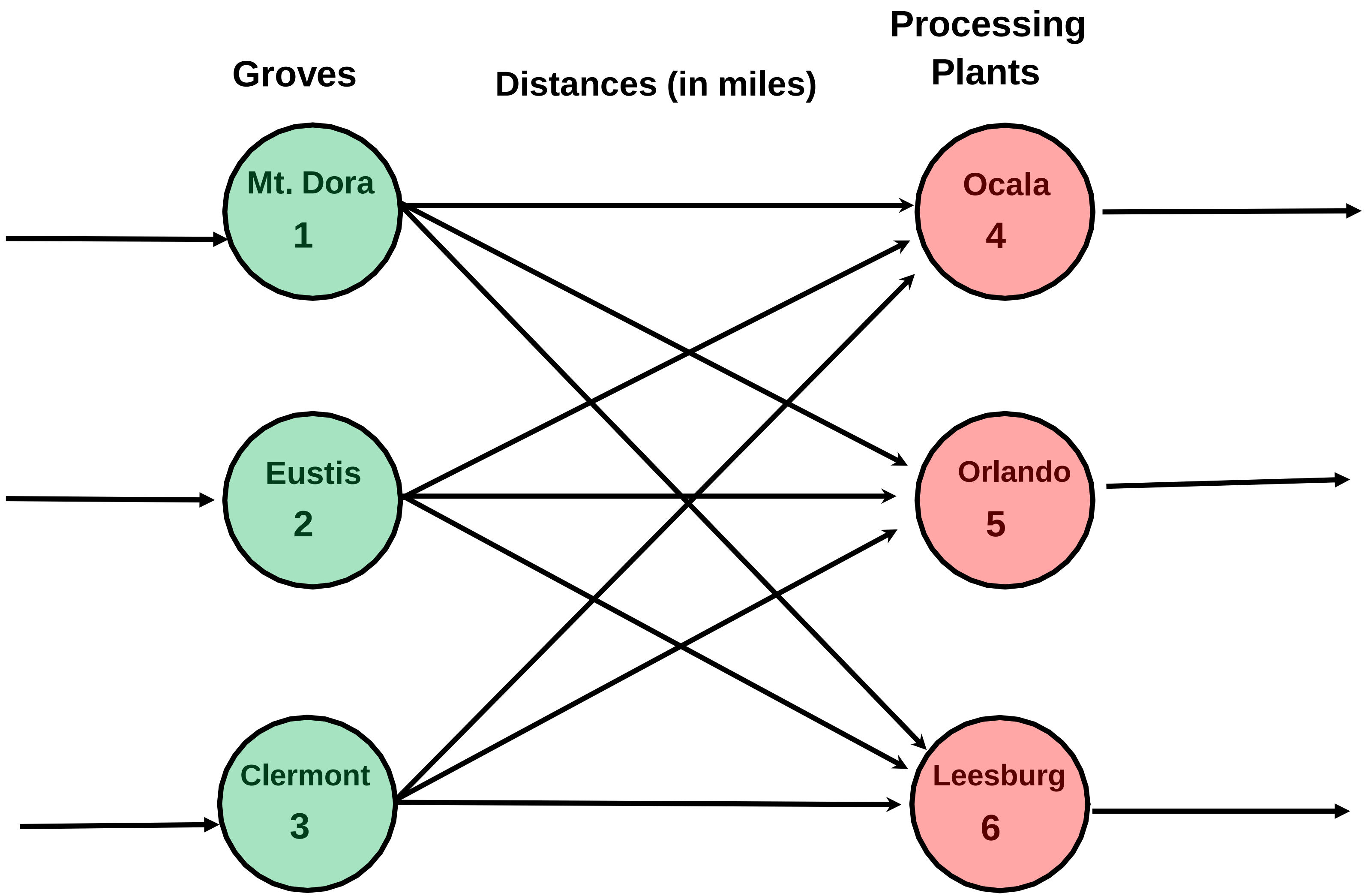
Grove	Supply
Mt. Dora	275,000
Eustis	400,000
Clermont	300,000

Plant	Capacity
Ocala	200,000
Orlando	600,000
Leesburg	225,000

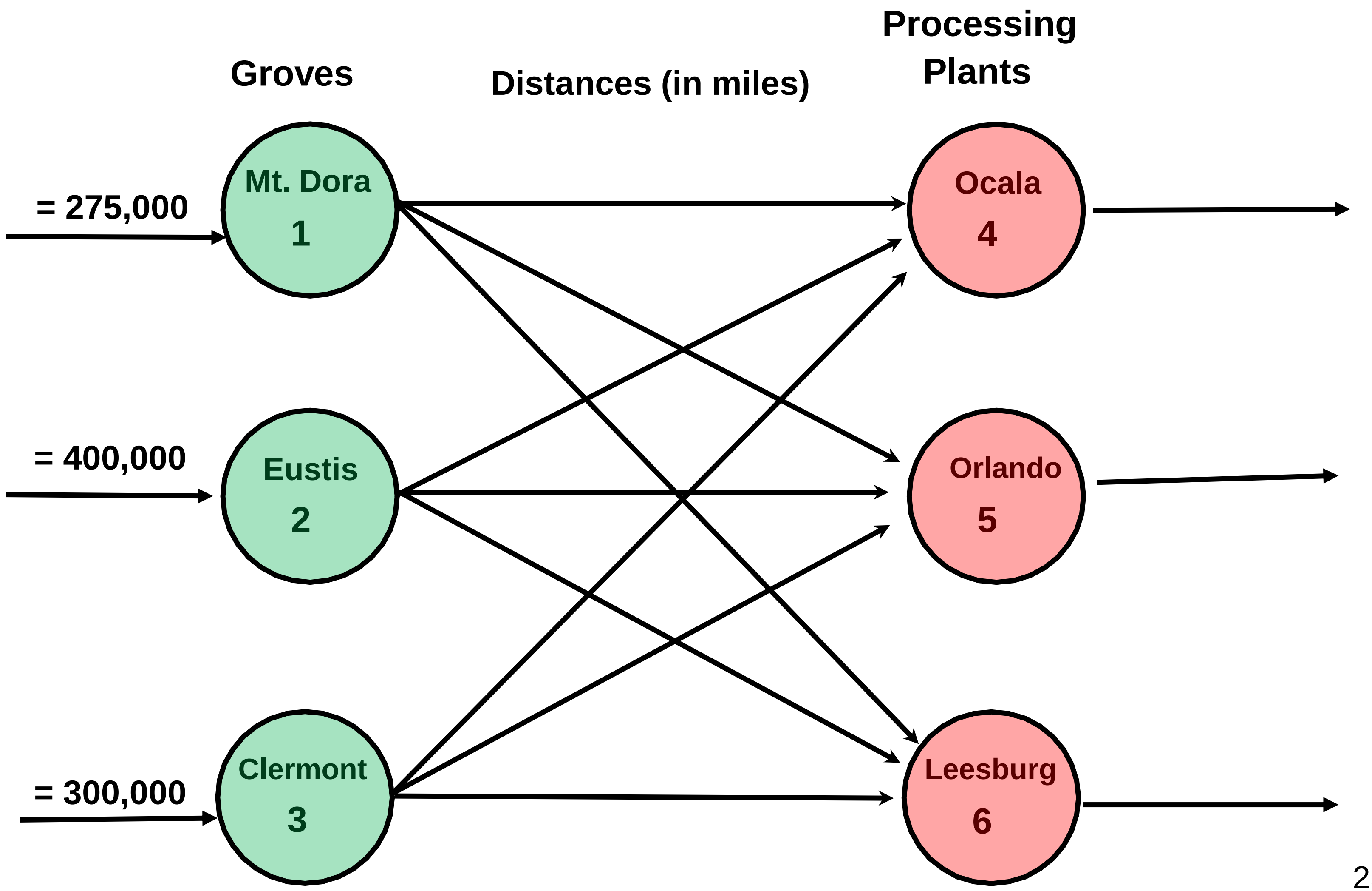
Distance	Ocala	Orlando	Leesburg
Mt. Dora	21	50	40
Eustis	35	30	22
Clermont	55	20	25

- How can we transport ***the entire*** supply of oranges from the groves to the plants such that the total distance is minimized?
- Formulate the transportation network and solve the LP model.

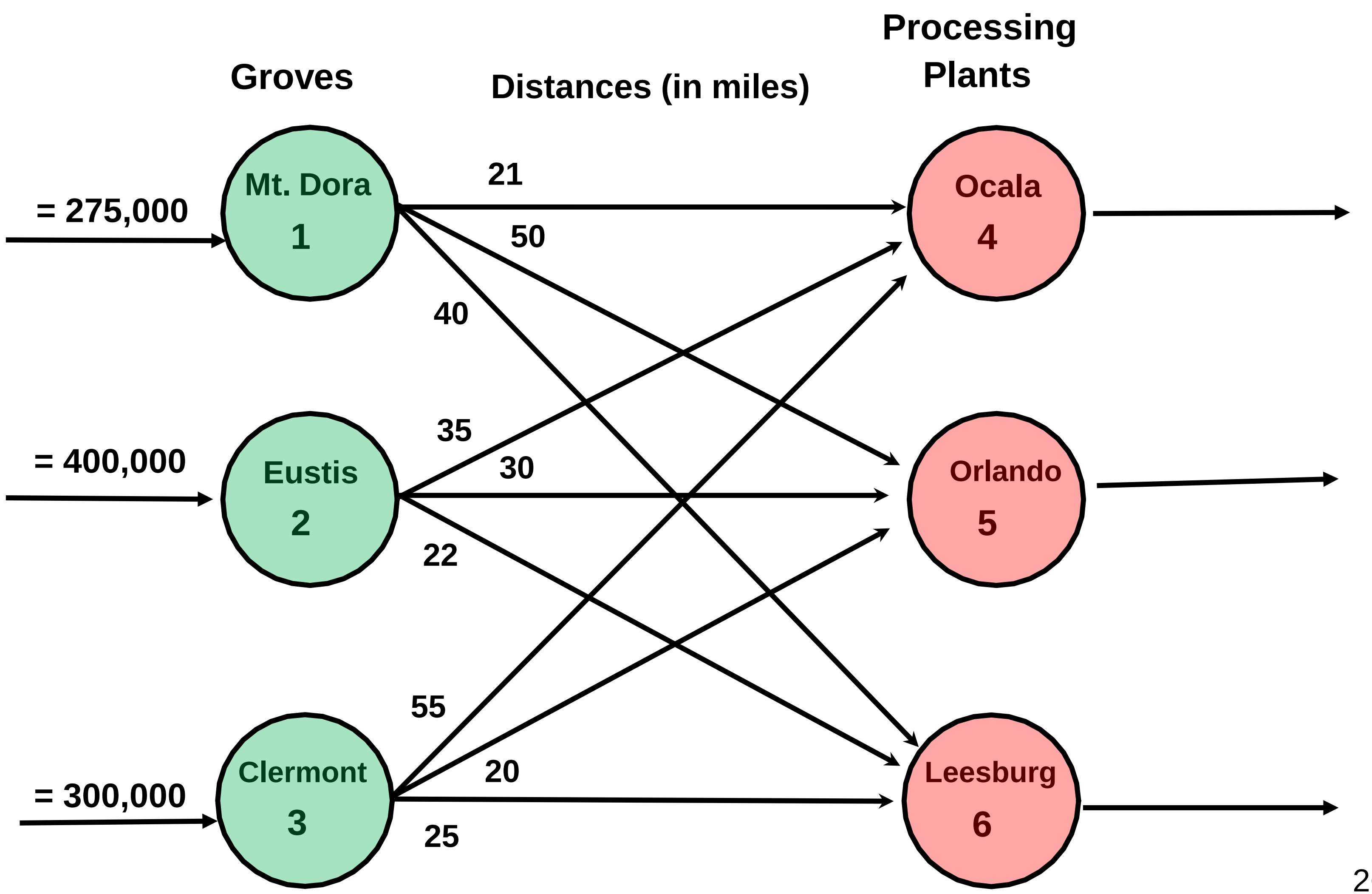
Transportation Example



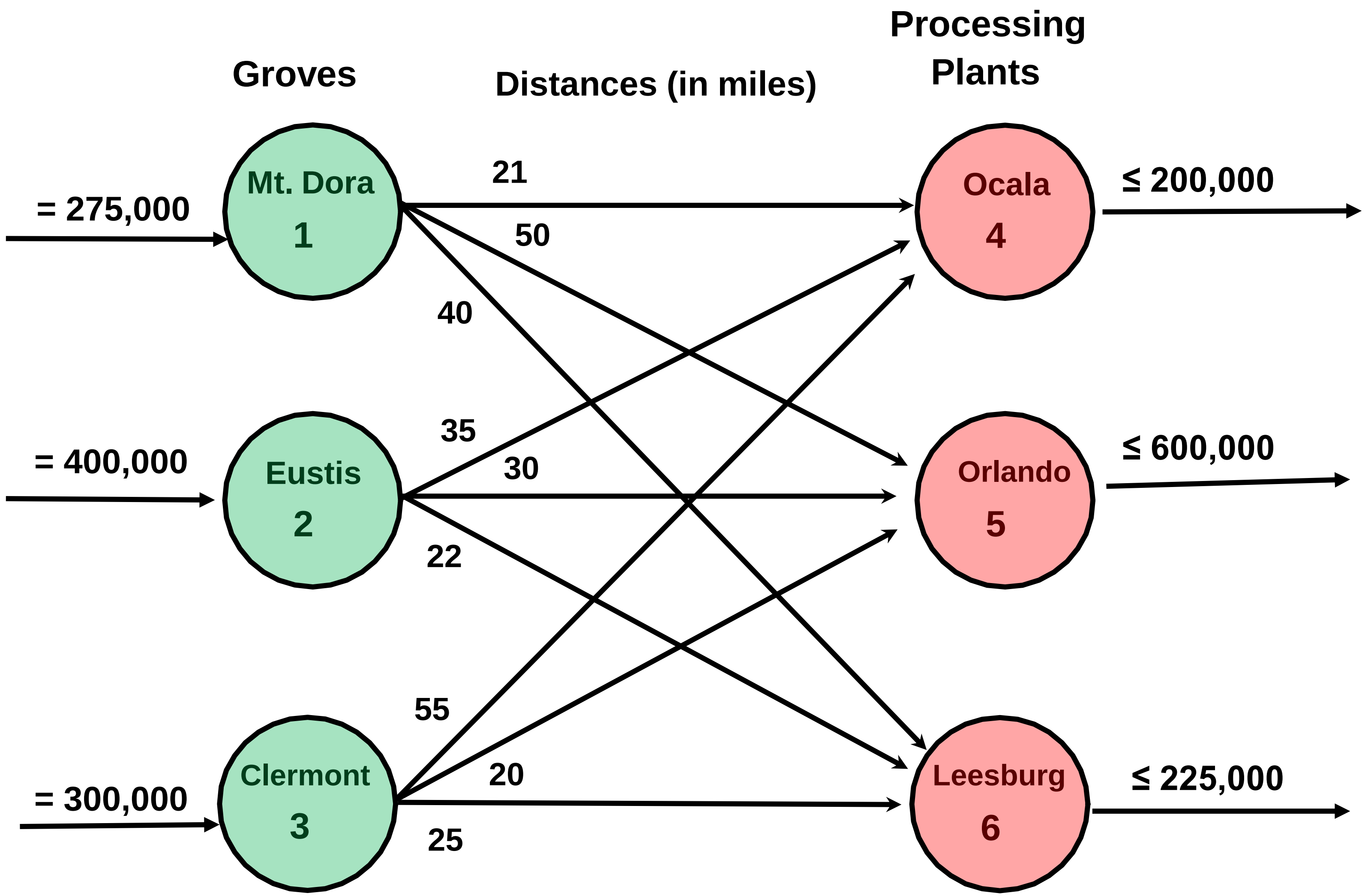
Transportation Example



Transportation Example



Transportation Example



Transportation Example

Define the objective:

Decision variables:

Write the mathematical objective function:

Transportation Example

Define the objective:

Minimize the total distance that <u>all</u> oranges travel

Decision variables:

Write the mathematical objective function:

Transportation Example

Define the objective:

Minimize the total distance that all oranges travel

Decision variables:

$\boldsymbol{x_{ij}}$: amount of oranges sent *from* node $\boldsymbol{i}$ *to* node $\boldsymbol{j}$
where $\boldsymbol{i} = \{\boldsymbol{Mt}, \boldsymbol{Eu}, \boldsymbol{Cl}\}$ *and* $\boldsymbol{j} = \{\boldsymbol{Oc}, \boldsymbol{Or}, \boldsymbol{Le}\}$.

Write the mathematical objective function:

Transportation Example

Define the objective:

Minimize the total distance that <u>all</u> oranges travel

Decision variables:

$\boldsymbol{x_{ij}}$: amount of oranges sent *<u>from</u>* node $\boldsymbol{i}$ *<u>to</u>* node $\boldsymbol{j}$

where $\boldsymbol{i} = \{\boldsymbol{Mt}, \boldsymbol{Eu}, \boldsymbol{Cl}\}$ *and* $\boldsymbol{j} = \{\boldsymbol{Oc}, \boldsymbol{Or}, \boldsymbol{Le}\}$.

Write the mathematical objective function:

Minimize $\boldsymbol{Z = 21x_{Mt,Oc} + 50x_{Mt,Or} + \ldots + 25x_{Cl,Le}}$

Transportation Example

Formulating the constraints

There are three types of constraints:

1. Supply constraints
2. Demand constraints
3. Non-negativity constraints

Transportation Example

Formulating the supply constraints

Transport the entire supply of oranges from the groves to the processing plants.

Transportation Example

Formulating the supply constraints

Transport the entire supply of oranges from the groves to the processing plants.

$$x_{Mt,Oc} + x_{Mt,Or} + x_{Mt,Le} = 275000$$
$$x_{Eu,Oc} + x_{Eu,Or} + x_{Eu,Le} = 400000$$
$$x_{Cl,Oc} + x_{Cl,Or} + x_{Cl,Le} = 300000$$

Transportation Example

Formulating the demand constraints

The processing plants can only accept so many oranges (they have a capacity limit).

Transportation Example

Formulating the demand constraints

The processing plants can only accept so many oranges (they have a capacity limit).

$$x_{Mt,Oc} + x_{Eu,Oc} + x_{Cl,Oc} \leq 200000$$

$$x_{Mt,Or} + x_{Eu,Or} + x_{Cl,Or} \leq 600000$$

$$x_{Mt,Le} + x_{Eu,Le} + x_{Cl,Le} \leq 225000$$

Transportation Example

Minimize $\boldsymbol{Z} = 21x_{Mt,Oc} + 50x_{Mt,Or} + \ldots + 25x_{Cl,Le}$

Subject to:

$x_{Mt,Oc} + x_{Mt,Or} + x_{Mt,Le} = 275000$ (Supply for Mt.Dora grove)

$x_{Eu,Oc} + x_{Eu,Or} + x_{Eu,Le} = 400000$ (Supply for Eustis grove)

$x_{Cl,Oc} + x_{Cl,Or} + x_{Cl,Le} = 300000$ (Supply for Clermont grove)

$x_{Mt,Oc} + x_{Eu,Oc} + x_{Cl,Oc} \leq 200000$ (Demand capacity of Ocala)

$x_{Mt,Or} + x_{Eu,Or} + x_{Cl,Or} \leq 600000$ (Demand capacity of Orlando)

$x_{Mt,Le} + x_{Eu,Le} + x_{Cl,Le} \leq 225000$ (Demand capacity of Leesburg)

$x_{Mt,Oc} \geq 0, \ldots, x_{Cl,Le} \geq 0$ (Non-negativity constraints)

Transportation Example: Python Solution

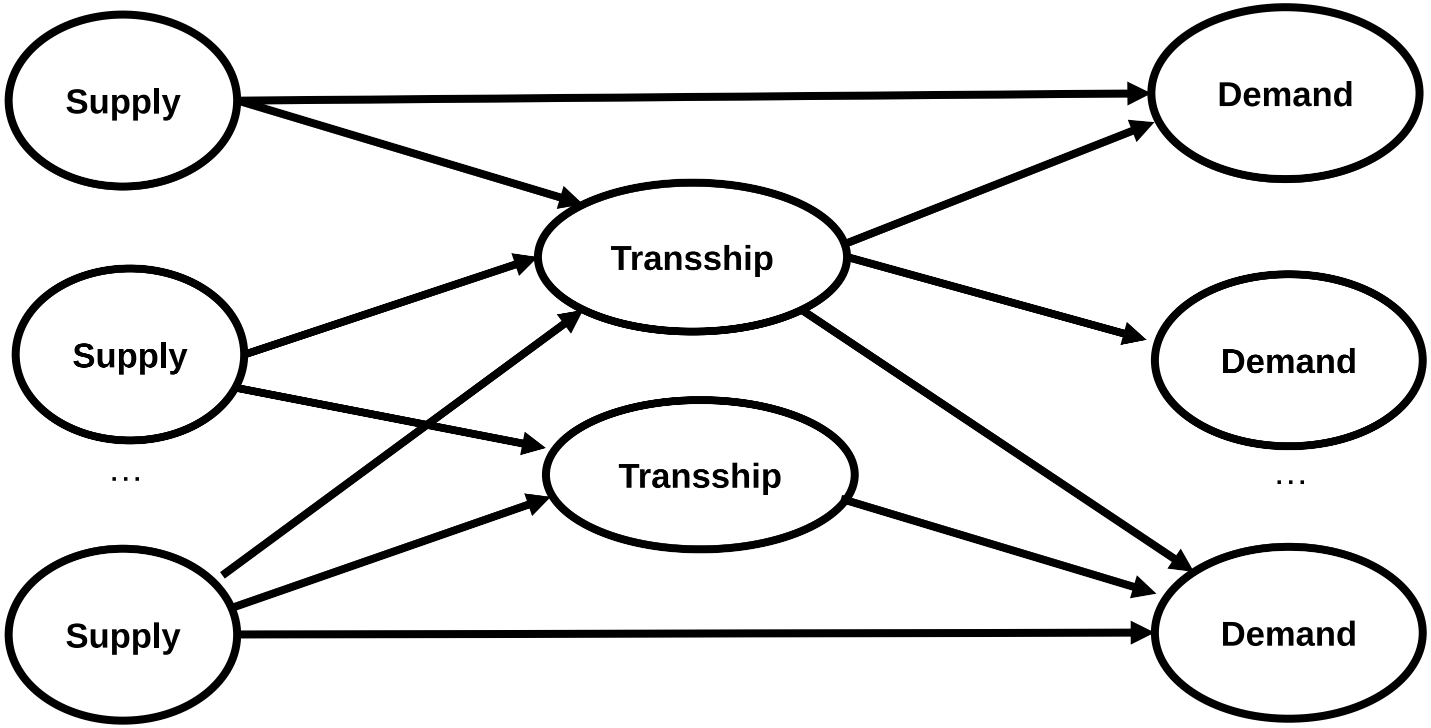
- Involves minimizing the cost of sending ***flow*** (e.g., people, products, oil) subject to:
  - Processing limitations, transportation capacity, management constraints, and both supply and demand restrictions.
- It is a very common optimization problem especially in supply chain management.

What managerial intuition do you get from the Python solution?

Transshipment Example

Transshipment Example

- Generalizes transportation problems by adding extra intermediate nodes called **transshipment nodes**.
  - Intermediate nodes may represent warehouses, cross-docking facilities, refinement or finishing centers, etc.



Transshipment Example

GE produces special digital components that are used in expensive electrical appliances like washers, dryers, microwaves, and dishwashers. They have two **plants**. One is in Dallas and the other is in Houston. GE ships the digital components to two **wholesalers** that distribute the products all over the world. The **wholesalers** are based in New York City and San Francisco. All products are shipped by air. Because of the pricing system of the airline, it can be cheaper to first ship to **intermediate** cities, Chicago or Los Angeles, rather than taking a direct flight. However, there is limited capacity to use these routes.

Transshipment Example

City	Max Production
Dallas	200
Houston	160

City	Capacity
Chicago	90
LA	80

City	Demand
SF	140
NY	140

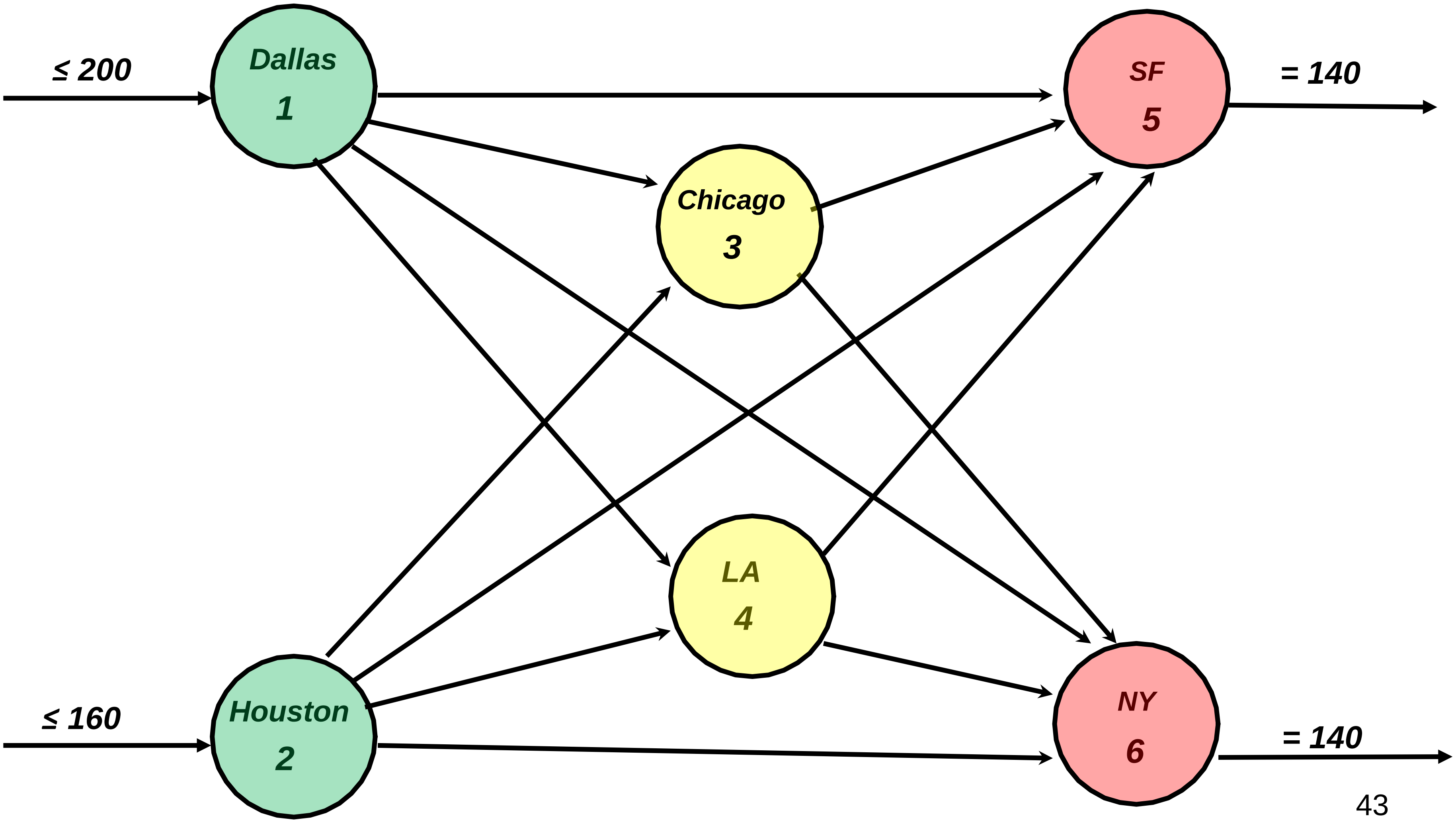
FROM	Dallas	Houston	Chicago	LA	SF	NY
Dallas	$0	-	$11	$10	$26	$29
Houston	-	$0	$9	$12	$27	$26
Chicago	-	-	$0	-	$12	$16
LA	-	-	-	$0	$13	$15
SF	-	-	-	-	$0	-
NY	-	-	-	-	-	$0

airfreight cost per unit

Production, capacity, and demand figures are in units of 1000.

How should GE transport products such that the shipping cost is minimized, the wholesalers are able to satisfy *all* of their demand, and no more than 40% of all products are shipped through Chicago and LA?

Transshipment Example



Transshipment Example

Define the objective:

Minimize the total shipping cost

Decision variables:

Write the mathematical objective function:

Transshipment Example

Define the objective:

Minimize the total shipping cost

Decision variables:

$\boldsymbol{x_{ij}}$: number of products sent <u>*from*</u> node $\boldsymbol{i}$ <u>*to*</u> node $\boldsymbol{j}$ where $\boldsymbol{i} = \{\boldsymbol{Da}, \boldsymbol{Ho}\}$ *and* $\boldsymbol{j} = \{\boldsymbol{Ch}, \boldsymbol{LA}, \boldsymbol{SF}, \boldsymbol{NY}\}$.

Write the mathematical objective function:

Transshipment Example

Define the objective:

Minimize the total shipping cost

Decision variables:

$\boldsymbol{y_{ij}}$: number of products sent *from* node $\boldsymbol{i}$ *to* node $\boldsymbol{j}$ where $\boldsymbol{i} = \{\boldsymbol{Ch}, \boldsymbol{LA}\}$ *and* $\boldsymbol{j} = \{\boldsymbol{SF}, \boldsymbol{NY}\}$.

Write the mathematical objective function:

Transshipment Example

Define the objective:

Minimize the total shipping cost

Decision variables:

$\boldsymbol{x_{ij}}$: number of products sent *<u>from</u>* node $\boldsymbol{i}$ <u>*to*</u> node $\boldsymbol{j}$ where $\boldsymbol{i} = \{\boldsymbol{Da}, \boldsymbol{Ho}\}$ *and* $\boldsymbol{j} = \{\boldsymbol{Ch}, \boldsymbol{LA}, \boldsymbol{SF}, \boldsymbol{NY}\}$.

$\boldsymbol{y_{ij}}$: number of products sent *<u>from</u>* node $\boldsymbol{i}$ <u>*to*</u> node $\boldsymbol{j}$ where $\boldsymbol{i} = \{\boldsymbol{Ch}, \boldsymbol{LA}\}$ *and* $\boldsymbol{j} = \{\boldsymbol{SF}, \boldsymbol{NY}\}$.

Write the mathematical objective function:

Transshipment Example

Define the objective:

Minimize the total shipping cost

Decision variables:

$\boldsymbol{x_{ij}}$: number of products sent <u>*from*</u> node $\boldsymbol{i}$ <u>*to*</u> node $\boldsymbol{j}$ where $\boldsymbol{i = \{Da, Ho\}}$ *and* $\boldsymbol{j = \{Ch, LA, SF, NY\}}$.

$\boldsymbol{y_{ij}}$: number of products sent <u>*from*</u> node $\boldsymbol{i}$ <u>*to*</u> node $\boldsymbol{j}$ where $\boldsymbol{i = \{Ch, LA\}}$ *and* $\boldsymbol{j = \{SF, NY\}}$.

Write the mathematical objective function:

Minimize $\boldsymbol{Z = 11x_{Da,Ch} + 10x_{Da,LA} + \ldots + 15y_{LA,NY}}$

Transshipment Example

Formulating the constraints

There are five types of constraints:

1. Supply constraints
2. Demand constraints
3. Transship constraints
4. Ratio constraint
5. Non-negativity constraints

Transshipment Example

Formulating the supply constraints

There is a capacity limit on how many units can be transported out of the production facilities in Dallas and Houston.

Transshipment Example

Formulating the supply constraints

There is a capacity limit on how many units can be transported out of the production facilities in Dallas and Houston.

$$x_{Da,Ch} + x_{Da,LA} + x_{Da,SF} + x_{Da,NY} \leq 200$$
$$x_{Ho,Ch} + x_{Ho,LA} + x_{Ho,SF} + x_{Ho,NY} \leq 160$$

Transshipment Example

Formulating the demand constraints

The wholesalers in San Francisco and New York must process all supply they receive.

Transshipment Example

Formulating the demand constraints

The wholesalers in San Francisco and New York must process all supply they receive.

$$x_{Da,SF} + x_{Ho,SF} + y_{Ch,SF} + y_{LA,SF} = 140$$
$$x_{Da,NY} + x_{Ho,NY} + y_{Ch,NY} + y_{LA,NY} = 140$$

Transshipment Example

Formulating the transship constraints

If products are shipped to Chicago and LA, they must satisfy the capacity constraints.

Transshipment Example

Formulating the transship constraints

If products are shipped to Chicago and LA, they must satisfy the capacity constraints.

$$x_{Da,Ch} + x_{Ho,Ch} \leq 90$$
$$x_{Da,LA} + x_{Ho,LA} \leq 80$$

Transshipment Example

Formulating the transship constraints

If products are shipped to Chicago and LA, they must eventually be distributed to San Francisco and New York.

Transshipment Example

Formulating the transship constraints

If products are shipped to Chicago and LA, they must eventually be distributed to San Francisco and New York.

$$x_{Da,Ch} + x_{Ho,Ch} = y_{Ch,SF} + y_{Ch,NY}$$
$$x_{Da,LA} + x_{Ho,LA} = y_{LA,SF} + y_{LA,NY}$$

Transshipment Example

Formulating the ratio constraint

No more than 40% of all products can be transshipped through Chicago and LA.

Transshipment Example

Formulating the ratio constraint

No more than 40% of all products can be transshipped through Chicago and LA.

$$\frac{y_{Ch,SF} + y_{Ch,NY} + y_{LA,SF} + y_{LA,NY}}{y_{Ch,SF} + y_{Ch,NY} + y_{LA,SF} + y_{LA,NY} + x_{Da,SF} + x_{Ho,SF} + x_{Da,NY} + x_{Ho,NY}} \leq 0.4$$

Transshipment Example

Formulating the ratio constraint

No more than 40% of all products can be transshipped through Chicago and LA.

$$0.6(y_{Ch,SF} + y_{Ch,NY} + y_{LA,SF} + y_{LA,NY}) \leq 0.4(x_{Da,SF} + x_{Ho,SF} + x_{Da,NY} + x_{Ho,NY})$$

Transshipment Example

Minimize $Z = 11x_{Da,Ch} + 10x_{Da,LA} + \ ... + 15y_{LA,NY}$

Subject to:

$x_{Da,Ch} + x_{Da,LA} + x_{Da,SF} + x_{Da,NY} \leq 200$ (Dallas production capacity)

$x_{Ho,Ch} + x_{Ho,LA} + x_{Ho,SF} + x_{Ho,NY} \leq 160$ (Houston production capacity)

$x_{Da,Ch} + x_{Ho,Ch} \leq 90$ (Chicago distribution capacity)

$x_{Da,LA} + x_{Ho,LA} \leq 80$ (LA distribution capacity)

$x_{Da,Ch} + x_{Ho,Ch} = y_{Ch,SF} + y_{Ch,NY}$ (Chicago flow constraint)

$x_{Da,LA} + x_{Ho,LA} = y_{LA,SF} + y_{LA,NY}$ (LA flow constraint)

$x_{Da,SF} + x_{Ho,SF} + y_{Ch,SF} + y_{LA,SF} = 140$ (SF demand constraint)

$x_{Da,NY} + x_{Ho,NY} + y_{Ch,NY} + y_{LA,NY} = 140$ (NY demand constraint)

$0.6(y_{Ch,SF} + y_{Ch,NY} + y_{LA,SF} + y_{LA,NY})$
$\leq 0.4(x_{Da,SF} + x_{Ho,SF} + x_{Da,NY} + x_{Ho,NY})$ (Ratio constraint)

$x_{Da,Ch} \geq 0, \ldots, y_{LA,NY} \geq 0$ (Non-negativity constraints)

Transshipment Example: Python Solution

- Notice how the constraints reflect the mathematical formulation of the problem.
  - This is not the case for all optimization solvers and is one reason why Gurobi is so popular.
- There is a difference between a variable in Python (*a memory location*) and a variable in an optimization model (*a decision variable that will be optimized by Gurobi*).

What managerial intuition do you get from the Python solution?

Tower Research® Example

Tower Research® Example

Tower Research® must define its investment portfolio for 4 months.

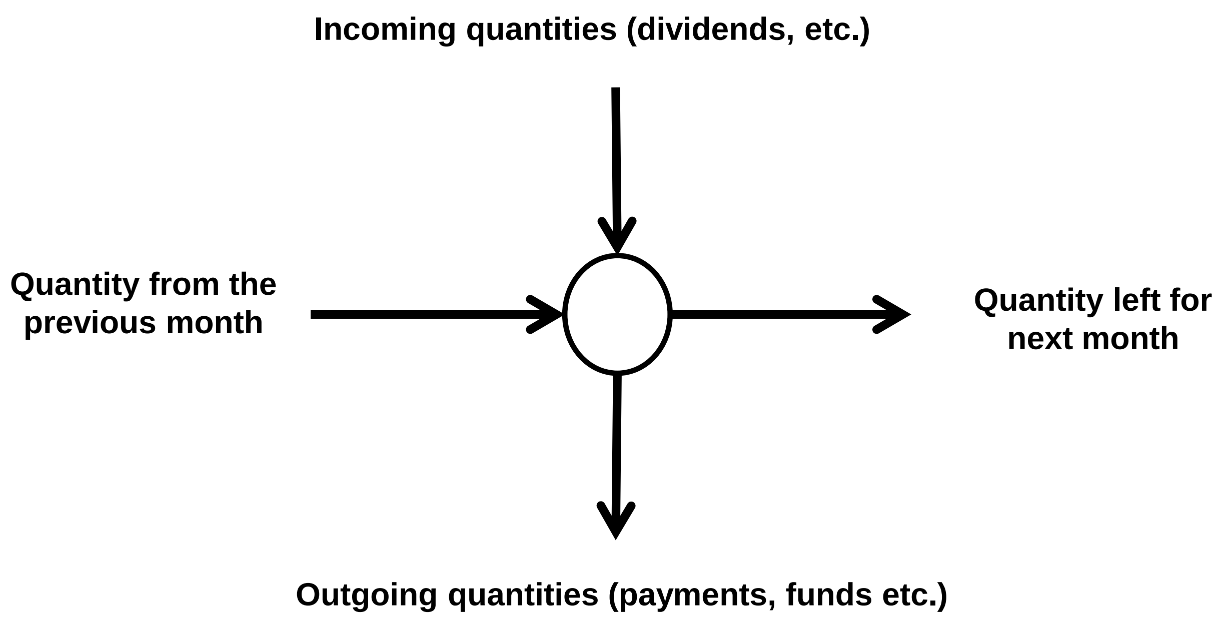
- It initially has $4,000 in deposits (figures in millions of dollars).
- After each month, the company has the option to invest in a ***2-month*** growth fund that gives a 2% (total) rate of return.
- The company can borrow, at most, $3,000 every month but must pay it back the ***following month*** at a 3% (total) rate.
- The expected investment revenues/expenses per month:

Month	1	2	3	4
Revenues	$1,000	$4,400	$5,800	$3,000
Expenses	$1,200	$4,800	$4,212	$1,000

What should Tower Research® do to maximize the amount that it generates at the end of the fourth month while never having a negative balance?

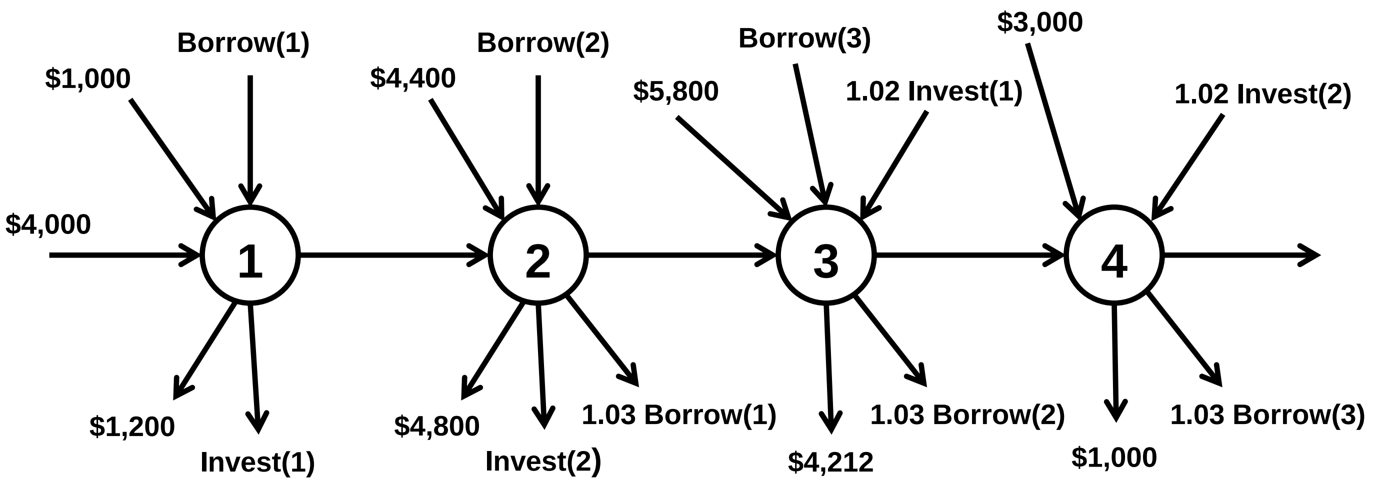
Tower Research® Example

Before solving multiperiod problems, it is a good practice to draw a **diagram** of the problem. For each month, we have:



Tower Research® Example

Four each of the four months:



Month	1	2	3	4
Revenues	$1,000	$4,400	$5,800	$3,000
Expenses	$1,200	$4,800	$4,212	$1,000

Tower Research® Example

Define the objective:

Maximize the amount of money at the end of period 4.

Decision variables:

Auxiliary decision variables:

Write the mathematical objective function:

Tower Research® Example

Define the objective:

Maximize the amount of money at the end of period 4.

Decision variables:

$I_t =$ how much we will **invest** in period $t = 1, 2$.

$B_t =$ how much we will **borrow** in period $t = 1, 2, 3$.

Auxiliary decision variables:

Write the mathematical objective function:

Tower Research® Example

Define the objective:

Maximize the amount of money at the end of period 4.

Decision variables:

I_t = how much we will **invest** in period $t = 1, 2$.

B_t = how much we will **borrow** in period $t = 1, 2, 3$.

Auxiliary decision variables:

w_t = total wealth **at the end** of period $t = 1, 2, 3, 4$.

Write the mathematical objective function:

Tower Research® Example

Define the objective:

Maximize the amount of money at the end of period 4.

Decision variables:

I_t = how much we will **invest** in period $t = 1, 2$.

B_t = how much we will **borrow** in period $t = 1, 2, 3$.

Auxiliary decision variables:

w_t = total wealth **at the end** of period $t = 1, 2, 3, 4$.

Write the mathematical objective function:

Maximize $\boldsymbol{Z = w_4}$

Tower Research® Example

Formulating the constraints

There are three types of constraints:

1. Borrowing constraints
2. Balance constraints
3. Non-negativity constraints

Tower Research® Example

Formulating the borrowing constraints

How do you ensure the company cannot borrow more than $3000 in any period?

Tower Research® Example

Formulating the borrowing constraints

How do you ensure the company cannot borrow more than $3000 in any period?

$$\boldsymbol{B_t \leq 3000 \; for \; t = 1, \ldots, 3}$$

Tower Research® Example

Formulating the balance constraints

- Ensure that the amount of money in each period is **consistent**. The amount at the end of a period is:
  1. The amount at the beginning of the period.
  2. Plus the deposits that are entering.
  3. Minus the withdrawals that are leaving.

- They are of the form:

$$\boldsymbol{w_t = w_{t-1} + inputs\ - outputs}$$

Tower Research® Example

Period 1:

Period 2:

Period 3:

Period 4:

Month	Start	1	2	3	4
Revenues	4000	$1,000	$4,400	$5,800	$3,000
Expenses	N/A	$1,200	$4,800	$4,212	$1,000

Tower Research® Example

Period 1:

$$w_1 = 4000 + 1000 + B_1 - 1200 - I_1$$

Period 2:

Period 3:

Period 4:

Month	Start	1	2	3	4
Revenues	4000	\$1,000	\$4,400	\$5,800	\$3,000
Expenses	N/A	\$1,200	\$4,800	\$4,212	\$1,000

Tower Research® Example

Period 1:

$$w_1 = 4000 + 1000 + B_1 - 1200 - I_1$$

Period 2:

$$w_2 = w_1 + B_2 + 4400 - 1.03B_1 - 4800 - I_2$$

Period 3:

Period 4:

Month	Start	1	2	3	4
Revenues	4000	\$1,000	\$4,400	\$5,800	\$3,000
Expenses	N/A	\$1,200	\$4,800	\$4,212	\$1,000

Tower Research® Example

Period 1:

$$w_1 = 4000 + 1000 + B_1 - 1200 - I_1$$

Period 2:

$$w_2 = w_1 + B_2 + 4400 - 1.03B_1 - 4800 - I_2$$

Period 3:

$$w_3 = w_2 + B_3 + 5800 + 1.02I_1 - 4212 - 1.03B_2$$

Period 4:

Month	Start	1	2	3	4
Revenues	4000	$1,000	$4,400	$5,800	$3,000
Expenses	N/A	$1,200	$4,800	$4,212	$1,000

Tower Research® Example

Period 1:

$$w_1 = 4000 + 1000 + B_1 - 1200 - I_1$$

Period 2:

$$w_2 = w_1 + B_2 + 4400 - 1.03B_1 - 4800 - I_2$$

Period 3:

$$w_3 = w_2 + B_3 + 5800 + 1.02I_1 - 4212 - 1.03B_2$$

Period 4:

$$w_4 = w_3 + 3000 + 1.02I_2 - 1000 - 1.03B_3$$

Month	Start	1	2	3	4
Revenues	4000	\$1,000	\$4,400	\$5,800	\$3,000
Expenses	N/A	\$1,200	\$4,800	\$4,212	\$1,000

Tower Research® Example

Maximize $\boldsymbol{Z} = w_4$

Subject to:

$w_1 = 4000 + 1000 + B_1 - 1200 - I_1$ (Balance constraint #1)

$w_2 = w_1 + B_2 + 4400 - 1.03B_1 - 4800 - I_2$ (Balance constraint #2)

$w_3 = w_2 + B_3 + 5800 + 1.02I_1 - 4212 - 1.03B_2$ (Balance constraint #3)

$w_4 = w_3 + 3000 + 1.02I_2 - 1000 - 1.03B_3$ (Balance constraint #4)

$B_t \leq 3000\ for\ t = 1, \dots, 3$ (Borrowing constraints)

$I_t \geq\ 0\ for\ t = 1, .., 2$ (Non-negativity constraints)

$B_t \geq\ 0\ for\ t = 1, .., 3$ (Non-negativity constraints)

$w_t \geq\ 0\ for\ t = 1, .., 4$ (Non-negativity constraints)

Tower Research® Example: Python Solution

- Be careful of keeping track of the variable indices and the mathematical notation.
- Linear programming models can be used to determine optimal decisions for multi-period problems. More complicated formulations can even include uncertainty.
  - **Examples:** Inventory planning, production scheduling, or workforce assignment.

What managerial intuition do you get from the Python solution?

Python Programming: Gurobi Cheat Sheet

Basic Functionality

Gurobi for Python

Modeling with Python and Gurobi:

Import and create a new instance of a Gurobi model:

```
import gurobipy as gb
from gurobipy import GRB
lp = gb.Model(“Name of Problem”)
```

Define a single decision variable:

```
x = model.addVar(<lb>, <ub>, <vtype>, “Name”)
```

- `<lb>` - Lower bound on value (e.g., lb=-`GRB.INFINITY, lb=0`).
- `<ub>` - Upper bound on value (e.g., ub=1, ub=`GRB.INFINITY).`
- `<vtype> - GRB.CONTINUOUS, GRB.BINARY, GRB.INTEGER`

Gurobi for Python

Modeling with Python and Gurobi:

Import and create a new instance of a Gurobi model:

```
import gurobipy as gb
from gurobipy import GRB
lp = gb.Model("Name of Problem")
```

Define multiple decision variables:

```
x = model.addVars(<num>, <lb>, <ub>, <vtype>, "Name")
```

- `<num>` - Number of decision variables (e.g., `100 or 2,100`).
- `<lb>` - Lower bound on value (e.g., lb=`-GRB.INFINITY, lb=0`).
- `<ub>` - Upper bound on value (e.g., ub=1, ub=`GRB.INFINITY)`.
- `<vtype> - GRB.CONTINUOUS, GRB.BINARY, GRB.INTEGER`

Gurobi for Python

Modeling with Python and Gurobi:

Add the objective to the Gurobi model:

```
lp = set.Objective(<function>, <sense>)
```

- Examples:

$$\text{lp.addConstr}(\text{x}[0] + x[1] + x[2] + x[3] + x[4], \text{GRB. MAXIMIZE})$$
$$\text{lp.addConstr}(\text{gb. quicksum}\big(\text{x[i] for } i \text{ in range}(4)\big), \text{GRB.MINIMIZE})$$

Add the constraints to the Gurobi model:

```
lp.addConstr(<equation>, “Name”)
```

- Examples:

$$\text{lp.addConstr}(\text{x}[0] + x[3] <= 10, \text{"Constraint"})$$
$$\text{lp.addConstr}(\text{gb. quicksum}\big(\text{x[i] for } i \text{ in range}(100)\big) >= 100, \text{"Cons"})$$

Gurobi for Python

Modeling with Python and Gurobi:

Solve the optimization problem:

```
lp.optimize()
```

Return the number of decision variables in the model:

```
lp.numVars
```

Return the number of constraints in the model:

```
lp.numConstrs
```

Return the status code of the model:

```
lp.status
```

Gurobi for Python

Modeling with Python and Gurobi:

Return optimal values from the solved model:

```
lp.printAttr('X') #print objective/optimal solution

x[0].x or x[1].x     #one-dimensional variables
x[0,0].x or x[0,1].x #two-dimensional variables
lp.getVars()     #returns all problem variables
```

Return the objective function value from the model:

```
lp.objVal
```

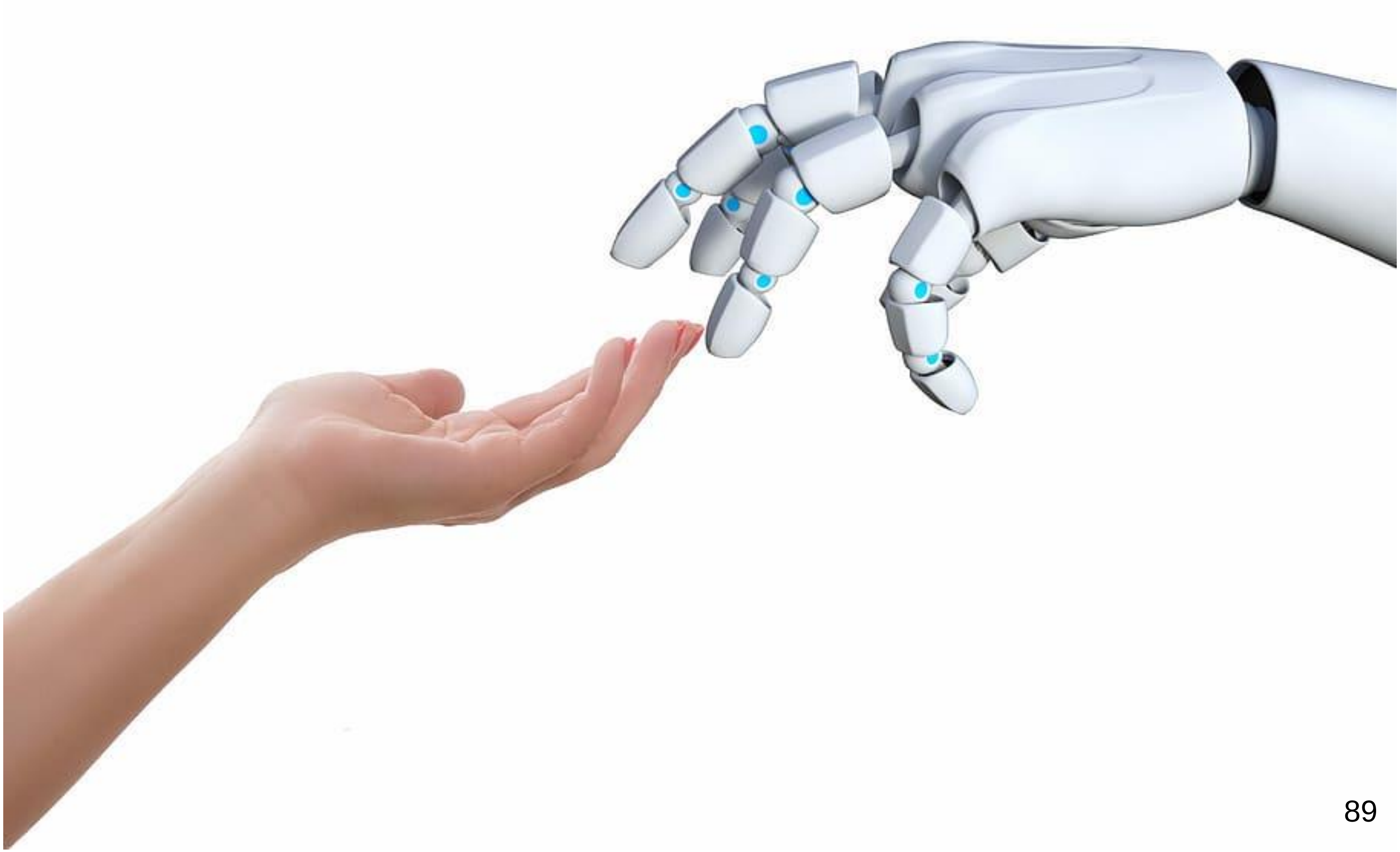
To see what else can be queried from the model, see this link.

Gurobi for Python

Modeling with Python and Gurobi:

- **Large-scale models:** In some instances, the solver may experience slow convergence toward the optimal solution, potentially taking hours or days to complete.
- Sometimes this can be rectified by **parameter tuning**.
- This is an advanced topic, see this webinar for details:
  - Set a time limit to solve the model. Analyze the optimality gap.
  - Control constraint tolerances (i.e., numerical precision).
  - Try a different optimization algorithm (e.g., concurrent optimizer).
  - Set the bounds of the decision variables upon instantiation.
  - Do not have large ranges in coefficient/parameter values.
  - Be careful when choosing certain parameter values.

Large Language Models (LLMs)

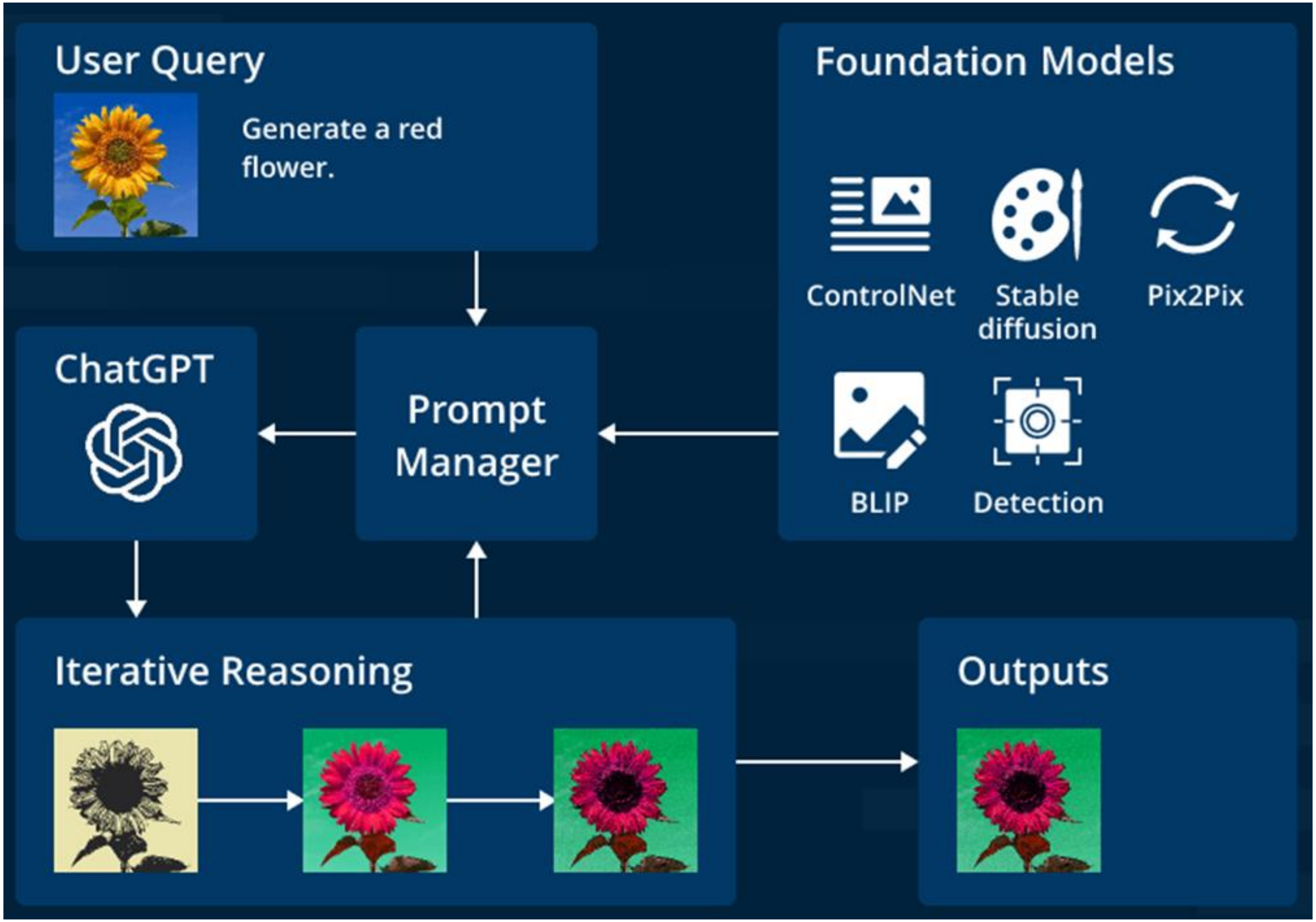


Large Language Models (LLMs)

Generative AI: Any set of artificial intelligence (AI) algorithms or tools that can *create* new content using pre-existing data (text, audio, video, images).

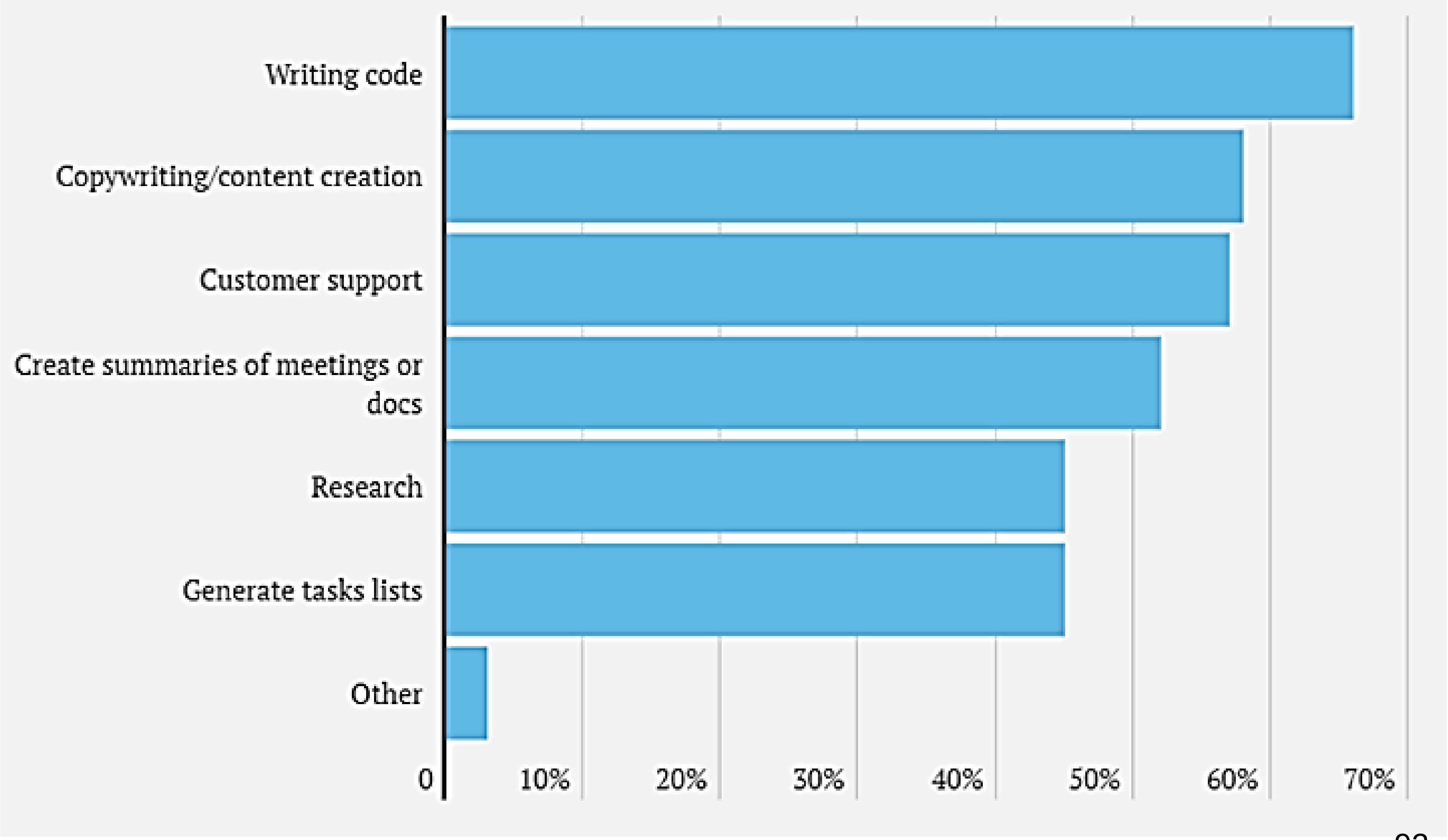
LLMs: AI systems that generate new content by interacting with users using written/spoken language.

Large Language Models (LLMs)



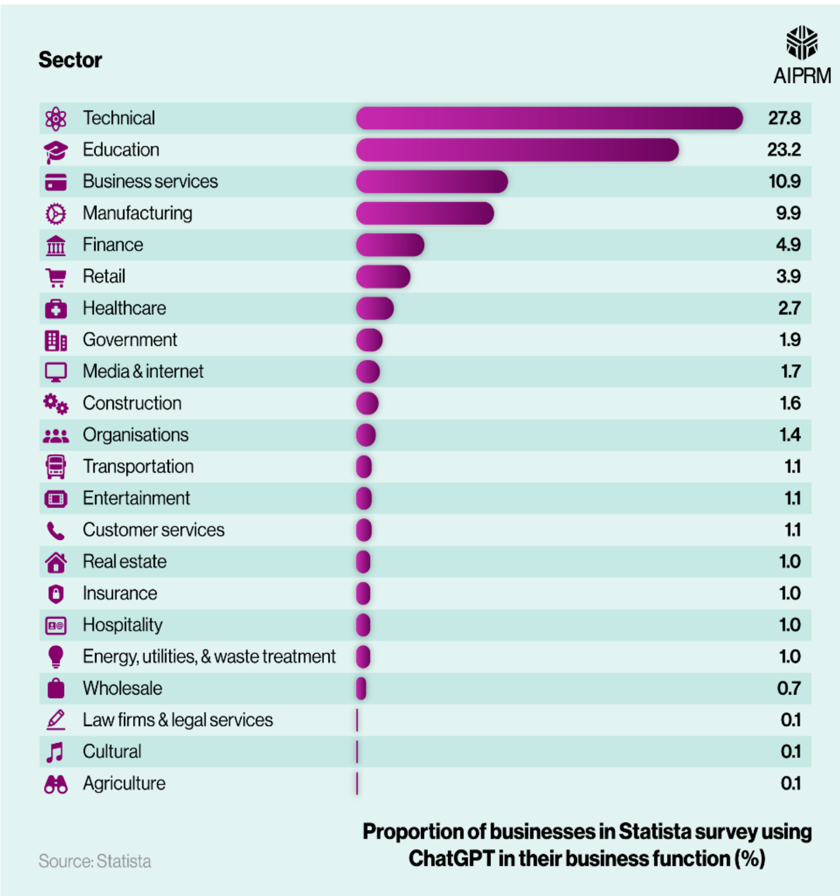
Source: https://www.leewayhertz.com/visual-chatgpt/

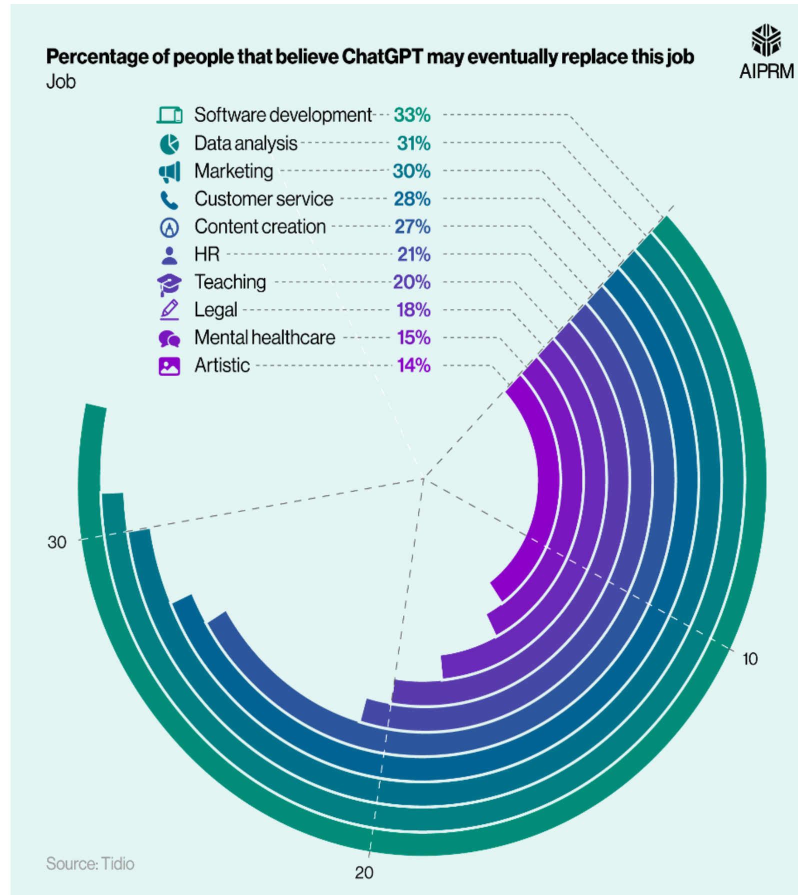
Large Language Models (LLMs)



Source: https://www.nextbigfuture.com/2023/04/1000-business-leaders-confirm-chatgpt-will-cause-job-loss-and-layoffs.html

ChatGPT ⊆ LLM





Source: https://www.aiprm.com/chatgpt-statistics/

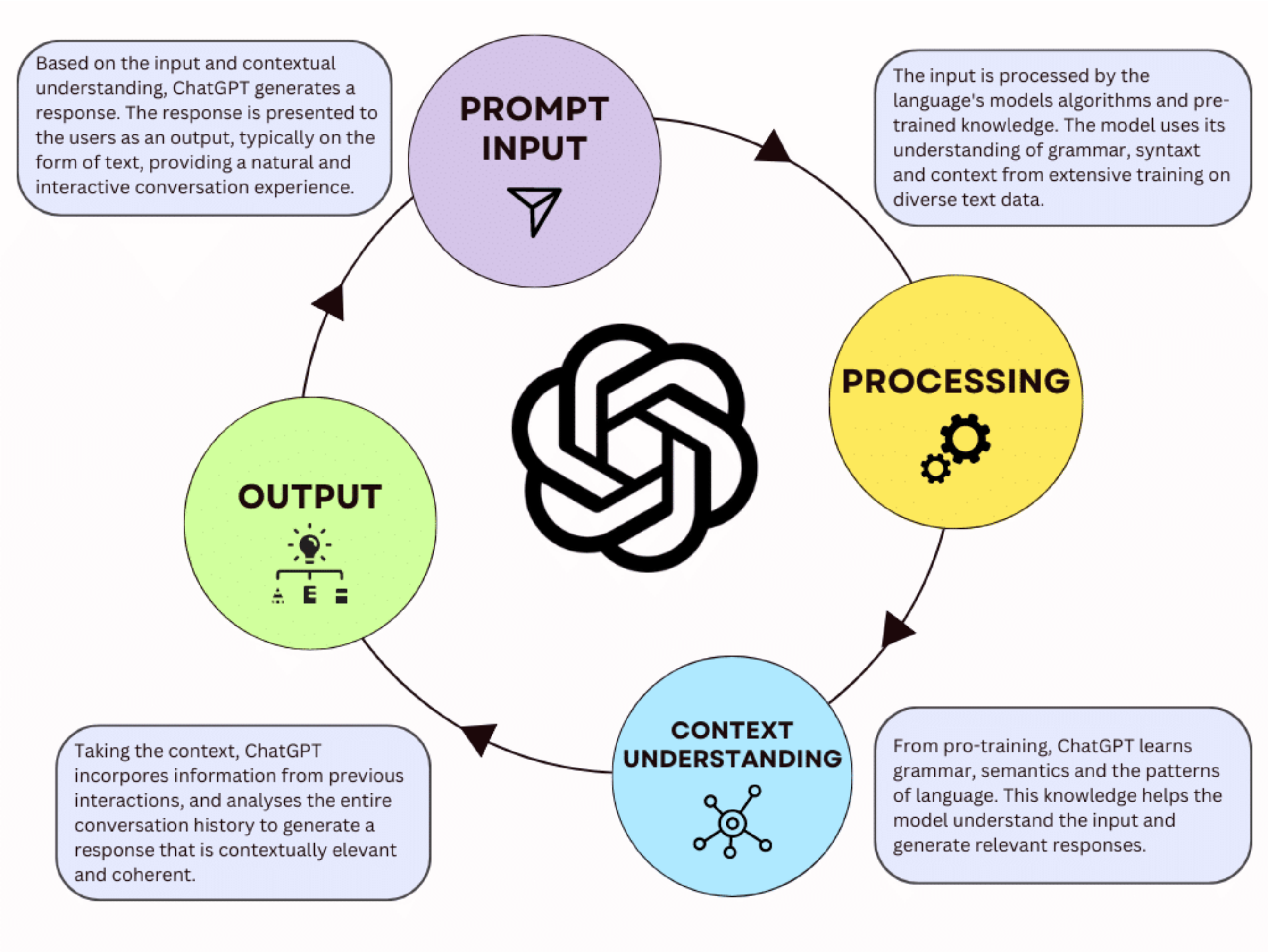
ChatGPT $\subseteq$ LLM

ChatGPT $\subseteq$ LLM

- Fundamentally, ChatGPT is an algorithm that predicts the most "*correct sounding*" answer to any question. It may not actually be correct…
- ChatGPT excels at generating lots of suggestions to answer any possible question. However, determining what the best solution is remains a uniquely human endeavor.
- The optimal workflow involves using ChatGPT as a personal assistant (known as a Co-Pilot).

"*It is both smarter and dumber than any person you've ever met.*"

ChatGPT ⊆ LLM



Source: *https://neuroflash.com/blog/chatgpt-wiki/ (spelling errors are not mine)*

ChatGPT ⊆ LLM

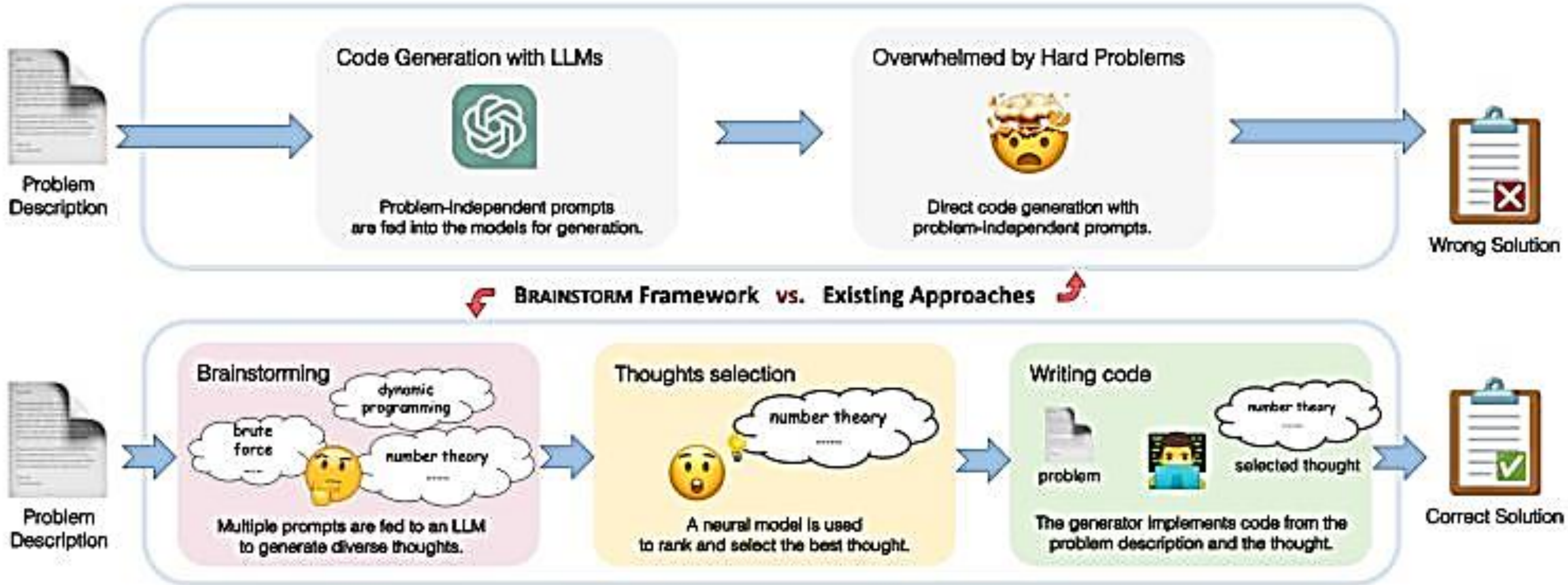
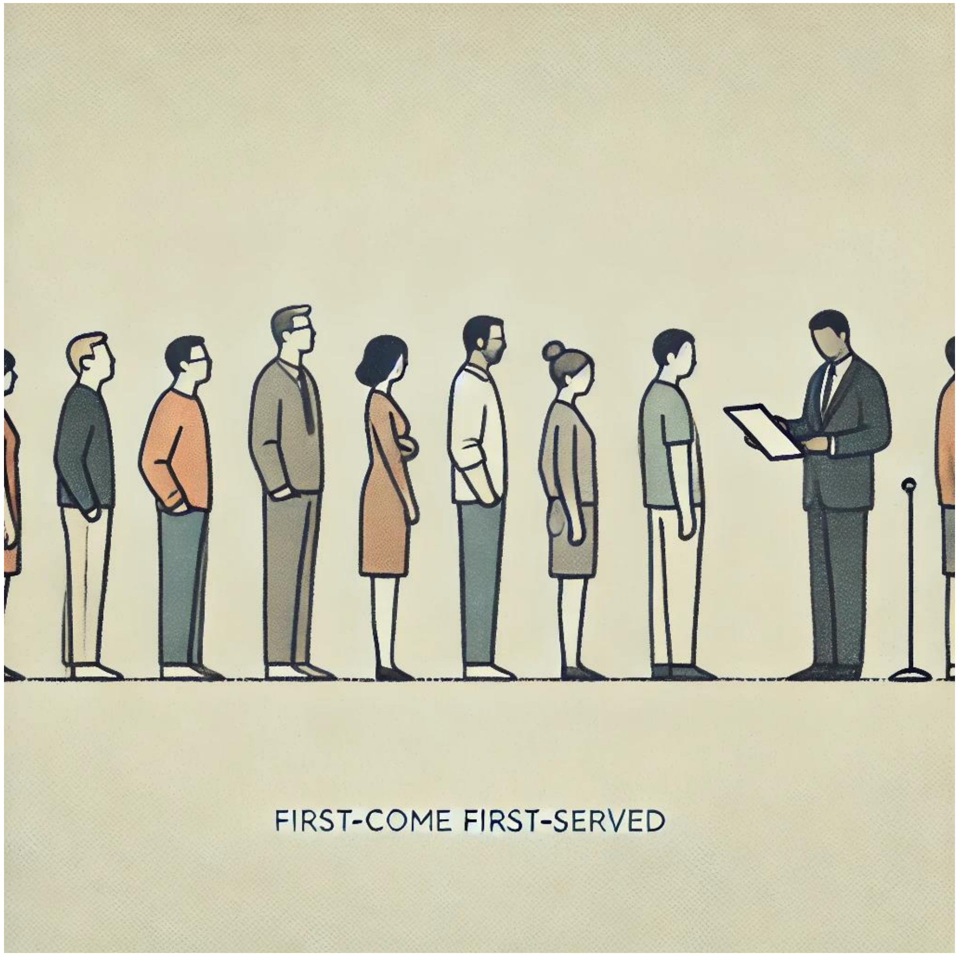


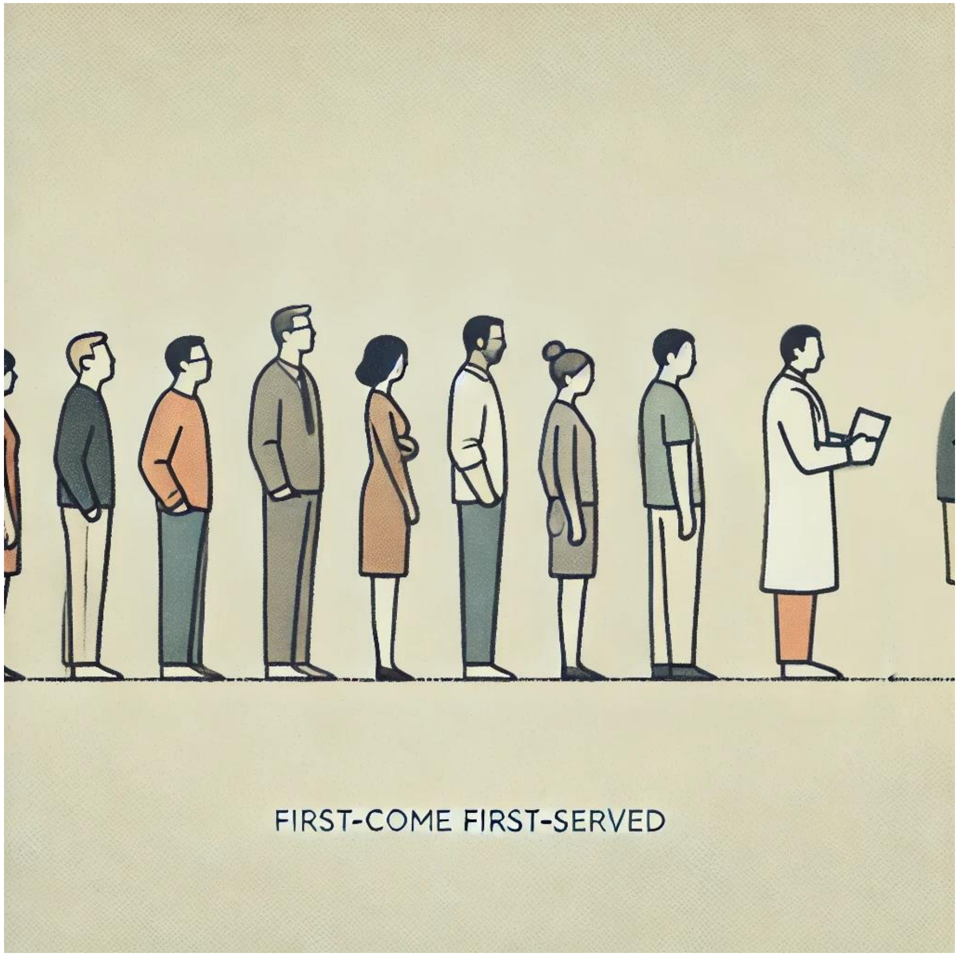
Figure 1: An illustrative example for the BRAINSTORM framework.

Source: https://arxiv.org/pdf/2305.10679v1.pdf

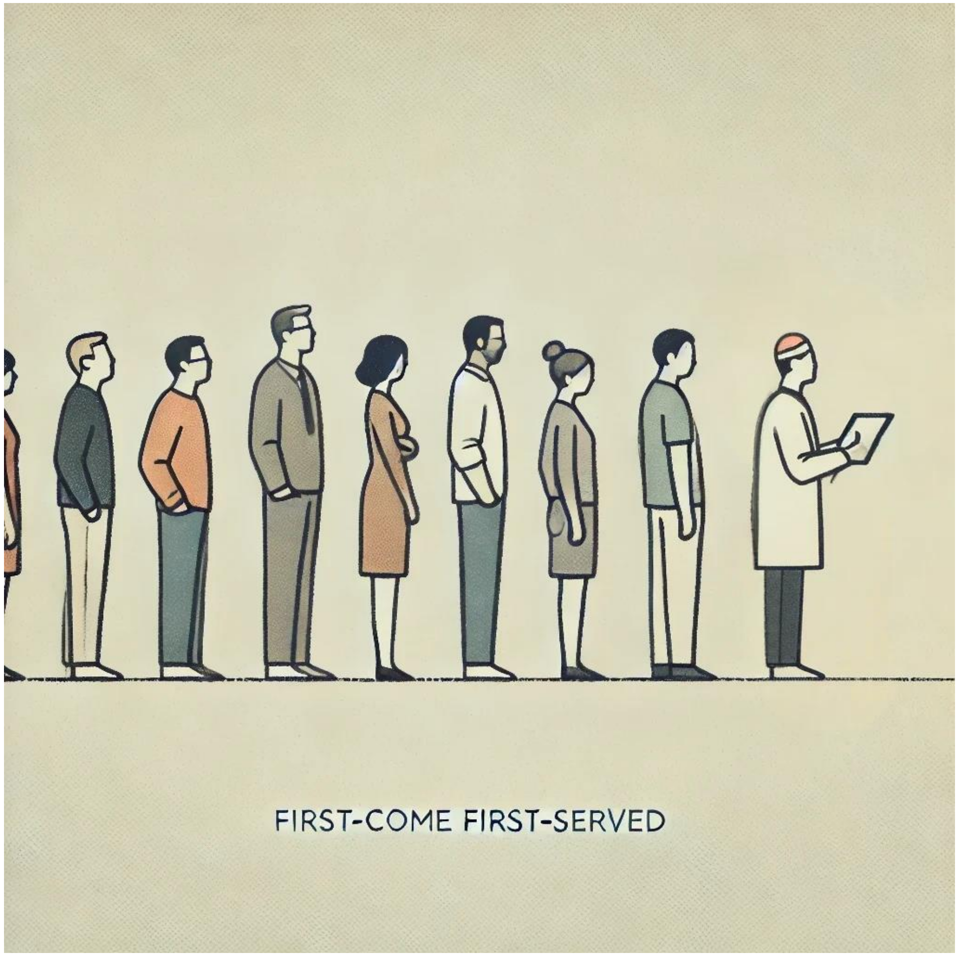
Prompt Engineering



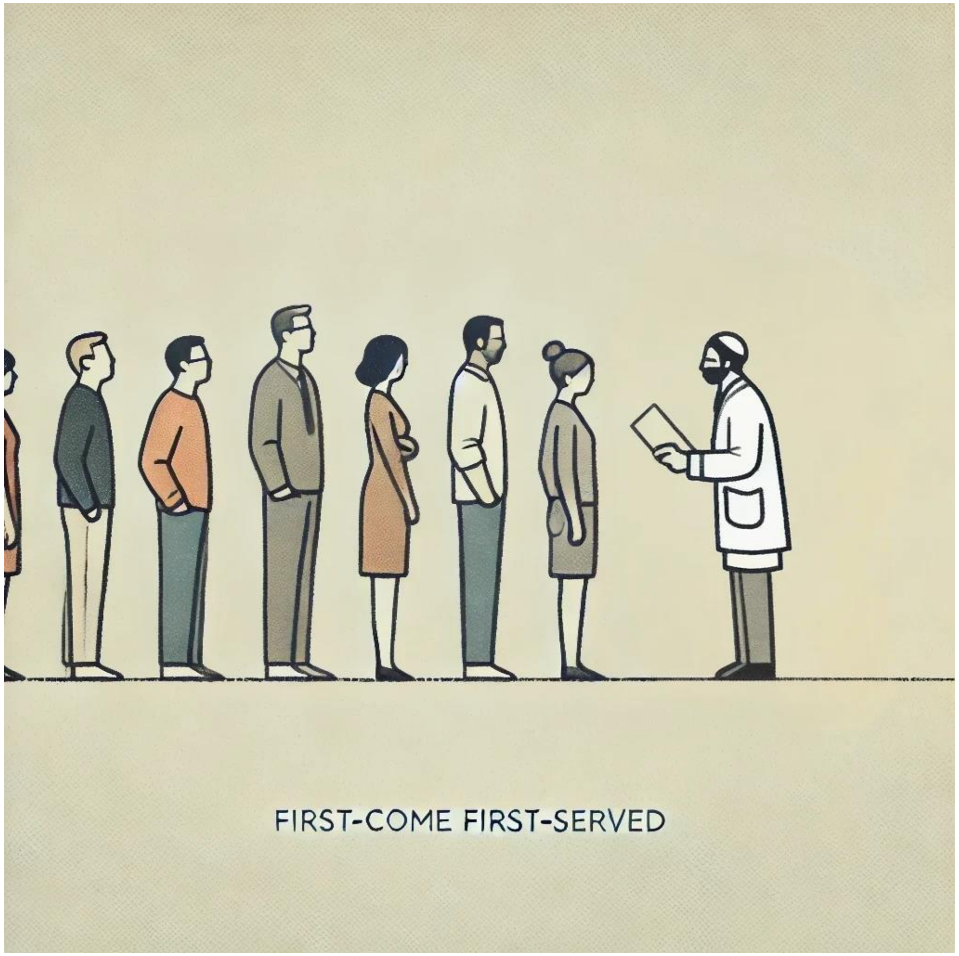
Prompt Engineering



Prompt Engineering



Prompt Engineering



Prompt Engineering

- You are the arbiter of truth!
  - AI tools do not know if they are right or wrong.
  - You need to determine whether the response is correct and useful. If not, rephrase your question or provide more context!
- When developing software, coders use unit testing to validate the expected behavior of the code. Many tests ensure that the code is correct in all cases.

Prompt Engineering

Verifying the correctness of an LLM answer:

- Regenerate the response to see if the LLM returns a different answer than the one it gave you.
  - Rephrase the question and ask it again.
- In more recent versions, ask the LLM to verify the information online and check the links it provides.
- Paste the solution in a new chat window and ask the LLM to evaluate inaccuracies in the answer.
- Provide contradictory statements to see if the LLM can identify which one is true and why.
- Analyze the response for "hedging" words like "may, might, possibly, potentially, uncertain…"

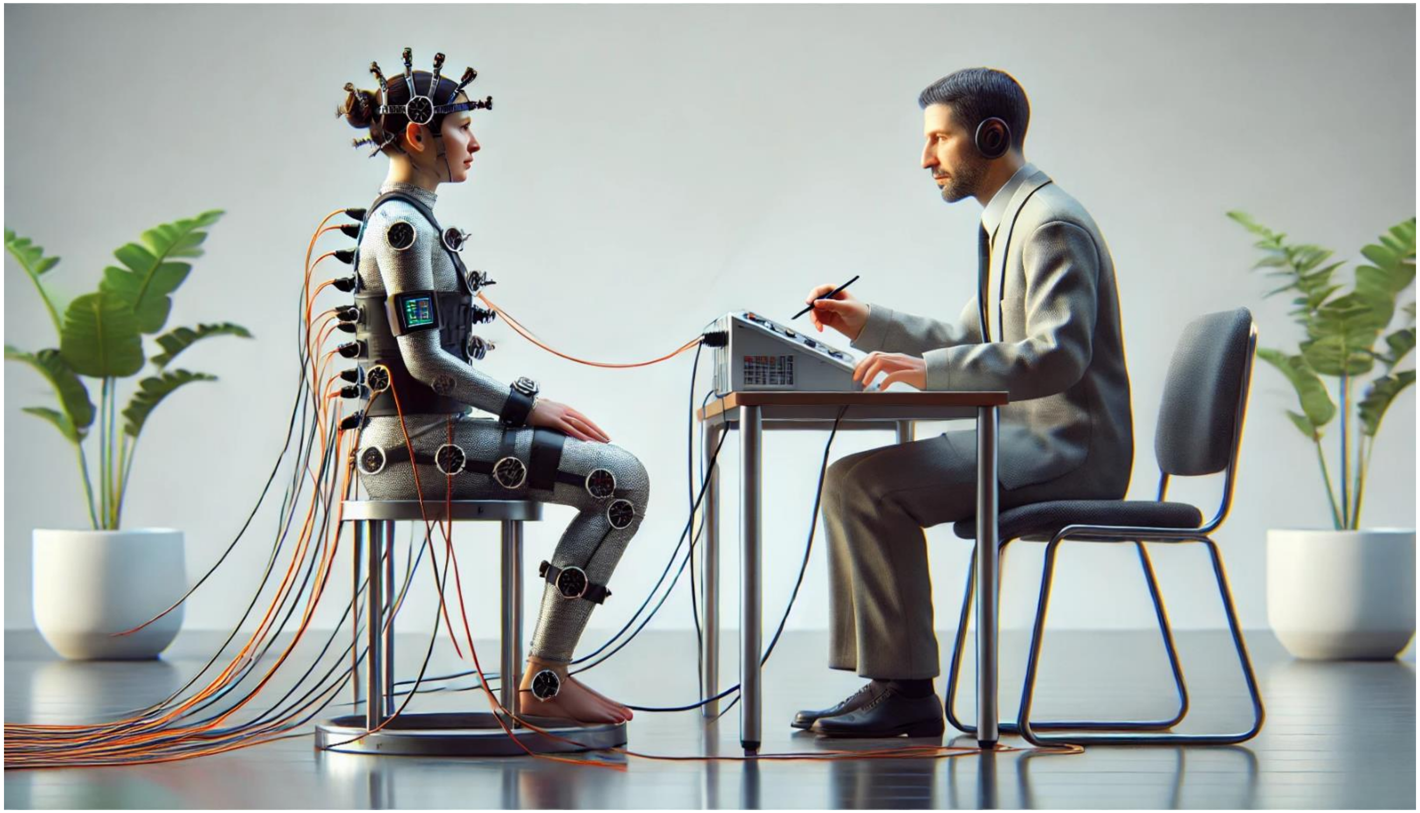
Prompt Engineering

Prompting tips and tricks:

- Show the LLM examples with the solution.
  - This technique is called multi-shot prompting.
- Decompose a large problem into simpler tasks where it is easier for you to verify correctness.
- Ask whether the proposed solution exhibits structure that is associated with a correct response.
- Use domain knowledge to identify issues in the solution. Explain to the LLM why the answer is incorrect and what a correct response looks like.
- Directly use the provided solution and modify it using your knowledge of the problem domain.

Prompt Engineering

1. Level-up your prompt engineering skills.
2. Deeply understand the problem domain.



Programming with ChatGPT

1. Understand the purpose of the code.
   – What are the objectives and data requirements?
2. Clearly define the inputs and outputs.
   – What is the data schema and where does it come from?
   – What information should the script return to the user?
3. Define the appropriate programming logic.
   – Do you want comments? Should the code be able to handle errors? How readable or performant should it be?
4. Test and refine the prompt/code.
   – Do not expect code for complex systems to automatically work. Always assume ChatGPT provides a "*first-draft*".

Programming with ChatGPT

Example: Define an LP using Python and gurobiypy

- Load a dataset from a csv file called "*Coefficients.csv*" that has the parameter values for the objective function.
- Maximize an objective function consisting of the sum of 20 continuous, non-negative decision variables where the first ten variables have an upper bound of 40 and the rest of the variables have an upper bound of 45.
- The sum of all decision variables is less than 100.
- Include constraints such that the sum of any three of the twenty decision variables is equal to no more than 66% of the total allocation in the optimal solution.

Programming with ChatGPT

```
import gurobipy as gp
import pandas as pd

# Load dataset from CSV file
dataset = pd.read_csv("Coefficients.csv")

# Create a new model
model = gp.Model("Linear Program")

# Create decision variables
variables = []
for i in range(20):
    if i < 10:
        ub = 40
    else:
        ub = 45
    variable = model.addVar(vtype=gp.GRB.CONTINUOUS, lb=0, ub=upper_bound)
    variables.append(variable)
```

Programming with ChatGPT

```
# Extract the coefficients for the objective function
objective_coefficients = coefficients_df.values.flatten()

# Set the objective function to maximize
model.setObjective(gp.quicksum(objective_coefficients[i] * variables[i] for
i in 20)), GRB.MAXIMIZE)

# Add the constraint: the sum of all decision variables is less than 100
model.addConstr(gp.quicksum(variables[i] for i in range(20)) <= 100)

# Add the constraint: the sum of any three variables is equal to no more
than 66% of the total allocation
for i in range(num_variables):
    for j in range(i + 1, num_variables):
        for k in range(j + 1, num_variables):
            model.addConstr(variables[i] + variables[j] + variables[k] <=
0.66 * gp.quicksum(variables), f"constraint_{i}_{j}_{k}")

model.optimize() # Optimize the model
```

Programming with ChatGPT

Using ChatGPT as a programming assistant is powerful!

- **Benefits:**
  - Quickly create a functional and moderately efficient codebase.
  - Can explain why/how certain snippets of code work.
  - Acts as a personal tutor/assistant for completing projects.
  - It can suggest alternative Python libraries and ways of coding.

- **Drawbacks:**
  - Requires you to clearly explain code functionality in great detail.
  - It sometimes hallucinates code/libraries that do not exist.
  - Does not tell you what logic may be missing from your code.
  - Typically requires extensive testing to ensure correctness for all use cases (although you can use ChatGPT to help with this).
  - There are persistent privacy and cybersecurity concerns.

Next Class: Linear Programming Duality

Different problem formulations that give identical optimal solutions! The difference is that the number of variables in the **dual problem** is equal to the number of constraints in the original (**primal**) problem. The number of constraints in the **dual problem** is equal to the number of variables in the **primal problem**. Both are linear programs!

- Weak and strong duality: Relating the objective functions and using them as optimality certificates.
- Reduced costs and shadow prices.
- Complementary slackness theory.