PC Part Optimizer

Ryan Chou

Introduction

Our optimization model is a PC part picker that selects the best parts (with "best" derived from individual performance metrics) for a custom gaming PC based on a user inputted budget, color, and RGB preference. The objective of the model is to maximize the performance of the PC while conforming to the user's provided preferences. The parts that the model will select for the PC are: GPU, CPU, motherboard, RAM, storage, case, CPU cooler, power supply, and case fans.

Motivation

As PC gaming has increased in popularity, the desire for gamers to build their own custom PCs as a cost effective method of obtaining a high-performance machine has also increased. However, a hurdle that many people face when seeking to build their own PC is deciding which products to choose and maneuvering the compatibility of different parts. The motivation behind our model comes from eliminating the lengthy process of research required for selecting proper pc parts. As long as a user has an idea of what they're looking for, the optimizer can produce a complete list of parts for them to purchase that will result in the highest performing PC possible.

Decision Variables

Each decision variable is a binary indicator representing whether a specific model of a certain PC part is to be included in the final build. The only exceptions are the integer decision variables for RAM and case fans, which represent the amount of each model to include. (See Appendix for link to spreadsheet that details which product each particular decision variable corresponds to.)

- $G_i \in B, i = 1, ..., 12$ (GPU models)
- $C_i \in B, i = 1, ..., 4$ (CPU models [Intel])
- $C_i \in B, i = 5, ..., 12$ (CPU models [AMD])
- $M_i \in B, i = 1, ..., 10$ (Motherboard models [Intel])
- $M_i \in B, i = 11, ..., 21$ (Motherboard models [AMD])
- $R_i \in B, i = 1, ..., 18$ (RAM models)
- $m_i \in Z, i = 1, ..., 18$ (1/2 No. of sticks for RAM model i)
- $S_i \in B, i = 1, ..., 5$ (Storage models [M.2SSD])
- $S_i \in B, i = 6, ..., 10$ (Storage models [2.5SSD])
- $S_i \in B, i = 11, ..., 15$ (Storage models [HDD])
- $E_i \in B, i = 1, ..., 16$ (Case models)
- $P_i \in B, i = 1, ..., 15$ (Power Supply models)
- $L_i \in B, i = 1, ..., 14$ (CPU Cooler models)
- $F_i \in B, i = 1, ..., 15$ (Case Fan models)
- $n_i \in Z, i = 1, ..., 15$ (No. of fans for Case Fan model i)

User Inputs

The user must select certain preferences and input them into the optimizer before running.

- **Color** (Black, White, Grey, Pink, Rainbow)
- **Budget**
- **RGB Lighting** (Yes, No, NA)

Additional Notes

To evaluate the performance of each product, we used different metrics for each type of PC part, and then standardized all performance metrics to be on a 1-10 scale. The metrics used to evaluate each part are as follows:

- GPU: Maximum FPS in Cyberpunk
- CPU: Base Frequency (GHz)
- Motherboard: Memory Max (GB)
- RAM: GB of Storage & DDR
- Storage: GB of Storage
- Case: Size (External Volume)
- Power Supply: Wattage
- Cooling: Idle CPU Temp over 21 Degrees C
- Fans: Max RPM

We also weighted the importance of each part's performance in the objective function, since some parts are more important for the overall quality of the PC than others. The weights are as follows:

- GPU: 1
- CPU: 1
- Motherboard: 1
- RAM: 0.8
- Storage: 0.6
- Case: 0.2
- Power Supply: 0.4
- Cooling: 0.6
- Fans: 0.2

Note About Code Screenshots: The code screenshots included in the report do not include preliminary variable declarations or other setup code. It only includes the lines of code that directly pertain to the explicit constraints listed. This was done in order to reduce the amount of space the screenshots took up since our full source code is very long. To view all of the code, please refer to the Github Repository in the Appendix.

Note About Model Scope: Due to limitations in the data we collected for our model, the optimizer doesn't currently provide an answer for every possible combination of Color and RGB. Given more time and resources, we would be able to add a larger variety of parts to the optimizer's list in order to ensure a solution for every possible combination of inputs. Currently, the optimizer will provide a complete solution for the following combinations: - Color: Black, RGB: Yes - Color: Black, RGB: No - Color: White, RGB: Yes - Color: Pink, RGB: Yes

Formulation

Objective Function

$$\begin{aligned}\max\ performance = &\sum_{i=1}^{12} PF_i G_i + \sum_{i=1}^{12} PF_i C_i + 0.8 \sum_{i=1}^{18} PF_i m_i \\ &+ \sum_{i=1}^{21} PF_i M_i + 0.6 \sum_{i=1}^{15} PF_i S_i + 0.2 \sum_{i=1}^{16} PF_i E_i \\ &+ 0.4 \sum_{i=1}^{15} PF_i P_i + 0.6 \sum_{i=1}^{14} PF_i L_i + 0.2 \sum_{i=1}^{15} PF_i n_i\end{aligned}$$

```python
# Objective
gpu_perf = sum(G[i] * gpu_performance[i] for i in gpu_indices)
cpu_perf = sum(C[i] * cpu_performance[i] for i in cpu_indices)
motherboard_perf = sum(M[i] * motherboard_performance[i] for i in motherboard_indices)
ram_perf = sum(2*m[i] * ram_performance[i] for i in ram_indices)
storage_perf = sum(S[i] * storage_performance[i] for i in storage_indices)
case_perf = sum(E[i] * case_performance[i] for i in case_indices)
ps_perf = sum(P[i] * ps_performance[i] for i in ps_indices)
cool_perf = sum(L[i] * cool_performance[i] for i in cool_indices)
fan_perf = sum(n[i] * fan_performance[i] for i in fan_indices)

model.setObjective(
    weights["GPU"] * gpu_perf
    + weights["CPU"] * cpu_perf
    + weights["motherboard"] * motherboard_perf
    + weights["RAM"] * ram_perf
    + weights["storage"] * storage_perf
    + weights["case"] * case_perf
    + weights["power supply"] * ps_perf
    + weights["cooling"] * cool_perf
    + weights["fans"] * fan_perf,
    GRB.MAXIMIZE
)
```

Price Constraint

$$\begin{aligned}&\sum_{i=1}^{12} Cost_i G_i + \sum_{i=1}^{12} Cost_i C_i + \sum_{i=1}^{18} Cost_i w_i \\ &+ \sum_{i=1}^{21} Cost_i M_i + \sum_{i=1}^{15} Cost_i S_i + \sum_{i=1}^{16} Cost_i E_i \\ &+ \sum_{i=1}^{15} Cost_i P_i + \sum_{i=1}^{14} Cost_i L_i + \sum_{i=1}^{15} Cost_i n_i \\ &\leq Price\end{aligned}$$

```python
# Budget constraint
model.addConstr(
    sum(G[i] * price_list[i] for i in gpu_indices)
    + sum(C[i] * cpu_price[i] for i in cpu_indices)
    + sum(M[i] * motherboard_price[i] for i in motherboard_indices)
    + sum(2*m[i] * ram_price[i] for i in ram_indices)
    + sum(S[i] * storage_price[i] for i in storage_indices)
    + sum(E[i] * case_price[i] for i in case_indices)
    + sum(P[i] * ps_price[i] for i in ps_indices)
    + sum(L[i] * cool_price[i] for i in cool_indices)
    + sum(n[i] * fan_price[i] for i in fan_indices)
    <= user_budget,
    "Budget_Limit"
)
```

GPU Constraints

$$\sum_{i=1}^{12} G_i = 1 \quad (Unique\ Constraint)$$

$$\sum_{i=1}^{12} Color_i G_i = Color \quad (Color\ Constraint)$$

$$(G_1 + G_5 + G_6 + G_8 + G_9 + G_{12}) - 1 \leq M(1 - z_1)$$
$$1 - (G_1 + G_5 + G_6 + G_8 + G_9 + G_{12}) \leq M(1 - z_1) \quad (RGB = Yes)$$
$$(G_2 + G_3 + G_4 + G_7 + G_{10} + G_{11}) - 1 \leq M(1 - z_0)$$
$$1 - (G_2 + G_3 + G_4 + G_7 + G_{10} + G_{11}) \leq M(1 - z_0) \quad (RGB = No)$$
$$(\sum_{i=1}^{12} G_i) - 1 \leq M(1 - z_2)$$
$$1 - (\sum_{i=1}^{12} G_i) \leq M(1 - z_2) \quad (RGB = NA)$$
$$z_0 + z_1 + z_2 = 1$$

$$z_0 = \begin{cases} 1 & \text{RGB=No} \\ 0 & \text{Otherwise} \end{cases} \quad z_1 = \begin{cases} 1 & \text{RGB=Yes} \\ 0 & \text{Otherwise} \end{cases} \quad z_2 = \begin{cases} 1 & \text{RGB=NA} \\ 0 & \text{Otherwise} \end{cases}$$

```
# GPU Constraints
# Note: Color constraint is bundled with other parts' color constraints below
model.addConstr(sum(G[i] for i in gpu_indices) == 1, "One_GPU")
# RGB Sets
G_rgb_yes = [i for i in gpu_indices if gpu_rgb[i] == 1]
G_rgb_no  = [i for i in gpu_indices if gpu_rgb[i] == 0]
# IF RGB Yes mode active
model.addConstr(sum(G[i] for i in G_rgb_yes) - 1 <= D * (1 - z1))
model.addConstr(1 - sum(G[i] for i in G_rgb_yes) <= D * (1 - z1))
# If RGB No mode active
model.addConstr(sum(G[i] for i in G_rgb_no) - 1 <= D * (1 - z0))
model.addConstr(1 - sum(G[i] for i in G_rgb_no) <= D * (1 - z0))
# If RGB NA
model.addConstr(sum(G[i] for i in gpu_indices) - 1 <= D * (1 - z2))
model.addConstr(1 - sum(G[i] for i in gpu_indices) <= D * (1 - z2))
```

CPU Constraints

$$\sum_{i=1}^{12} C_i = 1 \quad (Unique\ Constraint)$$

```
# CPU Constraints
model.addConstr(sum(C[i] for i in cpu_indices) == 1, "One_CPU")
```

Motherboard Constraints

$$\sum_{i=1}^{21} M_i = 1 \quad (Unique\ Constraint)$$

$$\sum_{i=1}^{21} Color_i M_i = Color \quad (Color\ Constraint)$$

$$\sum_{i=1}^{10} M_i \geq \sum_{i=1}^{4} C_i$$

$$\sum_{i=11}^{21} M_i \geq 1 - \sum_{i=1}^{4} C_i \quad (CPU/Motherboard\ Match)$$

```
# Motherboard Constraints
# Note: Color constraint is bundled with other parts' color constraints below
model.addConstr(sum(M[i] for i in motherboard_indices) == 1, "One_Motherboard")
#CPU and Motherboard compatibility
model.addConstr(sum(M[i] for i in range(0, 10)) >=  (C[0] + C[1] + C[2] + C[3]))
model.addConstr(sum(M[i] for i in range(10, 21)) >= 1 - (C[0] + C[1] + C[2] + C[3]))
```

RAM Constraints

$$\sum_{i=1}^{18} R_i = 1 \quad (Unique\ Constraint)$$

$$\sum_{i=1}^{18} Color_i R_i = Color \quad (Color\ Constraint)$$

$$(R_1 + R_4 + R_6 + R_8 + R_{12} + R_{14} + R_{15} + R_{17} + R_{18}) - 1 \leq M(1 - z_1)$$
$$1 - (R_1 + R_4 + R_6 + R_8 + R_{12} + R_{14} + R_{15} + R_{17} + R_{18}) \leq M(1 - z_1) \quad (RGB = Yes)$$
$$(R_2 + R_3 + R_5 + R_7 + R_9 + R_{10} + R_{11} + R_{13} + R_{16}) - 1 \leq M(1 - z_0)$$
$$1 - (R_2 + R_3 + R_5 + R_7 + R_9 + R_{10} + R_{11} + R_{13} + R_{16}) \leq M(1 - z_0) \quad (RGB = No)$$
$$(\sum_{i=1}^{18} R_i) - 1 \leq M(1 - z_2)$$
$$1 - (\sum_{i=1}^{18} R_i) \leq M(1 - z_2) \quad (RGB = NA)$$
$$z_0 + z_1 + z_2 = 1$$

$$For\ i = 1\ to\ 18: \quad 0 \leq m_i \leq 2R_i$$
$$w_i - 2(1 - R_i) \leq m_i \leq w_i$$
$$where\ w_i \in \{0, 1, 2\}$$
$$(Can\ Have\ 0, 2, or\ 4\ Sticks\ Constraint)$$

$$z_0 = \begin{cases} 1 & \text{RGB=No} \\ 0 & \text{Otherwise} \end{cases} \quad z_1 = \begin{cases} 1 & \text{RGB=Yes} \\ 0 & \text{Otherwise} \end{cases} \quad z_2 = \begin{cases} 1 & \text{RGB=NA} \\ 0 & \text{Otherwise} \end{cases}$$

```
# RAM Constraints
# Note: Color constraint is bundled with other parts' color constraints below
# Only 0, 2, 4 sticks allowed
w_max = 2
model.addConstr(sum(R[i] for i in ram_indices) == 1)
for i in ram_indices:
    model.addConstr(m[i] <= w_max * R[i], f"lin1_{i}")
    model.addConstr(m[i] >= 0, f"lin2_{i}")
    model.addConstr(m[i] <= w[i], f"lin3_{i}")
    model.addConstr(m[i] >= w[i] - w_max*(1-R[i]), f"lin4_{i}")
for i in ram_indices:
    model.addConstr(w[i] <= w_max * R[i], f"link_{i}")
```

Storage Constraints

$$\sum_{i=1}^{15} S_i \geq 1 \quad (At\ Least\ One\ Constraint)$$
$$\sum_{i=1}^{5} S_i \leq 1 \quad (Maximum\ 1\ Constraint)$$
$$\sum_{i=6}^{10} S_i \leq 1 \quad (Maximum\ 1\ Constraint)$$
$$\sum_{i=11}^{15} S_i \leq 1 \quad (Maximum\ 1\ Constraint)$$

```
# Storage Constraints
model.addConstr(sum(S[i] for i in storage_indices) >= 1, "One or more storage")
model.addConstr(sum(S[i] for i in range(0,5)) <= 1, "Max one M.2SSD")
model.addConstr(sum(S[i] for i in range(5,10)) <= 1, "Max one 2.5SSD")
model.addConstr(sum(S[i] for i in range(10,15)) <= 1, "Max one HDD")
```

Case Constraints

$$\sum_{i=1}^{16} E_i = 1 \quad (Unique\ Constraint)$$
$$\sum_{i=1}^{16} Color_i E_i = Color \quad (Color\ Constraint)$$
$$(E_1 + E_2 + E_5 + E_6 + E_7 + E_9 + E_{10} + E_{11} + E_{13} + E_{14} + E_{15} + E_{16}) - 1 \leq M(1 - z_1)$$
$$1 - (E_1 + E_2 + E_5 + E_6 + E_7 + E_9 + E_{10} + E_{11} + E_{13} + E_{14} + E_{15} + E_{16}) \leq M(1 - z_1) \quad (RGB = Yes)$$
$$(E_3 + E_4 + E_8 + E_{12}) - 1 \leq M(1 - z_0)$$
$$1 - (E_3 + E_4 + E_8 + E_{12}) \leq M(1 - z_0) \quad (RGB = No)$$
$$(\sum_{i=1}^{16} E_i) - 1 \leq M(1 - z_2)$$
$$1 - (\sum_{i=1}^{16} E_i) \leq M(1 - z_2) \quad (RGB = NA)$$
$$z_0 + z_1 + z_2 = 1$$

$$z_0 = \begin{cases} 1 & \text{RGB=No} \\ 0 & \text{Otherwise} \end{cases} \quad z_1 = \begin{cases} 1 & \text{RGB=Yes} \\ 0 & \text{Otherwise} \end{cases} \quad z_2 = \begin{cases} 1 & \text{RGB=NA} \\ 0 & \text{Otherwise} \end{cases}$$

```
# Case Constraints
# Note: Color constraint is bundled with other parts' color constraints below
model.addConstr(sum(E[i] for i in case_indices) == 1, "One_Case")
# RGB Sets
C_rgb_yes = [i for i in case_indices if case_rgb[i] == 1]
C_rgb_no  = [i for i in case_indices if case_rgb[i] == 0]
# If RGB Yes mode active
model.addConstr(sum(E[i] for i in C_rgb_yes) - 1 <= D * (1 - z1))
model.addConstr(1 - sum(E[i] for i in C_rgb_yes) <= D * (1 - z1))
# If RGB No mode active
model.addConstr(sum(E[i] for i in C_rgb_no) - 1 <= D * (1 - z0))
model.addConstr(1 - sum(E[i] for i in C_rgb_no) <= D * (1 - z0))
# If RGB NA
model.addConstr(sum(E[i] for i in case_indices) - 1 <= D * (1 - z2))
model.addConstr(1 - sum(E[i] for i in case_indices) <= D * (1 - z2))
```

Power Supply Constraints

$$\sum_{i=1}^{15} P_i = 1 \quad (Unique\ Constraint)$$

$$\sum_{i=1}^{15} Color_i P_i = Color \quad (Color\ Constraint)$$

```
# Power Supply Constraints
# Note: Color constraint is bundled with other parts' color constraints below
model.addConstr(sum(P[i] for i in ps_indices) == 1, "One_Power_supply")
```

Cooling Constraints

$$\sum_{i=1}^{14} L_i = 1 \quad (Unique\ Constraint)$$

$$\sum_{i=1}^{14} Color_i L_i = Color \quad (Color\ Constraint)$$

$$(L_4 + L_5 + \sum_{i=7}^{14} L_i) - 1 \leq M(1 - z_1)$$

$$1 - (L_4 + L_5 + \sum_{i=7}^{14} L_i) \leq M(1 - z_1) \quad (RGB = Yes)$$

$$(L_1 + L_2 + L_3 + L_6) - 1 \leq M(1 - z_0)$$

$$1 - (L_1 + L_2 + L_3 + L_6) \leq M(1 - z_0) \quad (RGB = No)$$

$$(\sum_{i=1}^{14} L_i) - 1 \leq M(1 - z_2)$$

$$1 - (\sum_{i=1}^{14} L_i) \leq M(1 - z_2) \quad (RGB = NA)$$

$$z_0 + z_1 + z_2 = 1$$

$$z_0 = \begin{cases} 1 & \text{RGB=No} \\ 0 & \text{Otherwise} \end{cases} \quad z_1 = \begin{cases} 1 & \text{RGB=Yes} \\ 0 & \text{Otherwise} \end{cases} \quad z_2 = \begin{cases} 1 & \text{RGB=NA} \\ 0 & \text{Otherwise} \end{cases}$$

```
# Cooling Constraints
# Note: Color constraint is bundled with other parts' color constraints below
model.addConstr(sum(L[i] for i in cool_indices) == 1, "One_Cool")
# RGB Sets
L_rgb_yes = [i for i in cool_indices if cool_rgb[i] == 1]
L_rgb_no  = [i for i in cool_indices if cool_rgb[i] == 0]
# If RGB Yes mode active
model.addConstr(sum(L[i] for i in L_rgb_yes) - 1 <= D * (1 - z1))
model.addConstr(1 - sum(L[i] for i in L_rgb_yes) <= D * (1 - z1))
# If RGB No mode active
model.addConstr(sum(L[i] for i in L_rgb_no) - 1 <= D * (1 - z0))
model.addConstr(1 - sum(L[i] for i in L_rgb_no) <= D * (1 - z0))
# If RGB NA
model.addConstr(sum(L[i] for i in cool_indices) - 1 <= D * (1 - z2))
model.addConstr(1 - sum(L[i] for i in cool_indices) <= D * (1 - z2))
```

Fans Constraints

$$\sum_{i=1}^{15} F_i \leq 1 \quad (Unique\ Constraint)$$

$$\sum_{i=1}^{15} Color_i F_i \leq Color + M(1 - \sum_{i=1}^{15} F_i)$$

$$\sum_{i=1}^{15} Color_i F_i \leq Color - M(1 - \sum_{i=1}^{15} F_i) \quad (Color\ Constraint)$$

$$(F_1 + F_3 + F_5 + F_6 + F_9 + F_{14} + F_{15}) - 1 \leq M(1 - z_1)$$

$$1 - (F_1 + F_3 + F_5 + F_6 + F_9 + F_{14} + F_{15}) \leq M(1 - z_1) \quad (RGB = Yes)$$

$$(F_2 + F_4 + F_7 + F_8 + F_{10} + F_{11} + F_{12} + F_{13}) - 1 \leq M(1 - z_0)$$

$$1 - (F_2 + F_4 + F_7 + F_8 + F_{10} + F_{11} + F_{12} + F_{13}) \leq M(1 - z_0) \quad (RGB = No)$$

$$(\sum_{i=1}^{15} F_i) - 1 \leq M(1 - z_2)$$

$$1 - (\sum_{i=1}^{15} F_i) \leq M(1 - z_2) \quad (RGB = NA)$$

$$z_0 + z_1 + z_2 = 1$$

$$For\ i = 1\ to\ 15: \quad 0 \leq n_i \leq 8F_i, \quad n_i \in \mathbb{Z}_{\geq 0}$$

$$where\ n_i = number\ of\ fans\ of\ fan\ type\ F_i$$

$$z_0 = \begin{cases} 1 & \text{RGB=No} \\ 0 & \text{Otherwise} \end{cases} \quad z_1 = \begin{cases} 1 & \text{RGB=Yes} \\ 0 & \text{Otherwise} \end{cases} \quad z_2 = \begin{cases} 1 & \text{RGB=NA} \\ 0 & \text{Otherwise} \end{cases}$$

```
# Fan Constraints
# Note: Color constraint is bundled with other parts' color constraints below
model.addConstr(sum(F[i] for i in fan_indices) == 1, "One_Fan_Type")
# RGB Sets
F_rgb_yes = [i for i in fan_indices if fan_rgb[i] == 1]
F_rgb_no  = [i for i in fan_indices if fan_rgb[i] == 0]
# If RGB Yes mode active
model.addConstr(sum(F[i] for i in F_rgb_yes) - 1 <= D * (1 - z1))
model.addConstr(1 - sum(F[i] for i in F_rgb_yes) <= D * (1 - z1))
# If RGB No mode active
model.addConstr(sum(F[i] for i in F_rgb_no) - 1 <= D * (1 - z0))
model.addConstr(1 - sum(F[i] for i in F_rgb_no) <= D * (1 - z0))
# If RGB NA
model.addConstr(sum(F[i] for i in fan_indices) - 1 <= D * (1 - z2))
model.addConstr(1 - sum(F[i] for i in fan_indices) <= D * (1 - z2))
# Can choose 0-8 case fans
f_max = 8
for i in fan_indices:
    model.addConstr(n[i] <= f_max * F[i], f"lin1_{i}")
    model.addConstr(n[i] >= 0, f"lin2_{i}")
    model.addConstr(n[i] <= f[i], f"lin3_{i}")
    model.addConstr(n[i] >= f[i] - f_max*(1-F[i]), f"lin4_{i}")
for i in fan_indices:
    model.addConstr(n[i] <= f_max * F[i], f"link_{i}")
```

Results/Post Optimality Analyses

The overall model works as intended and main aspects of the model (the budget and resulting performance) are reflected in the numbers correctly. As budget increases, the ability to use better (more expensive) parts becomes available, and overall performance increases as intended. Different budgets also lead to changes in parts picked to match the new budget. The output of our model provides a selected model for each PC part, along with the number of RAM sticks and case fans to include in the built. It also displays the cost of the build and the performance score. An example output is as follows:

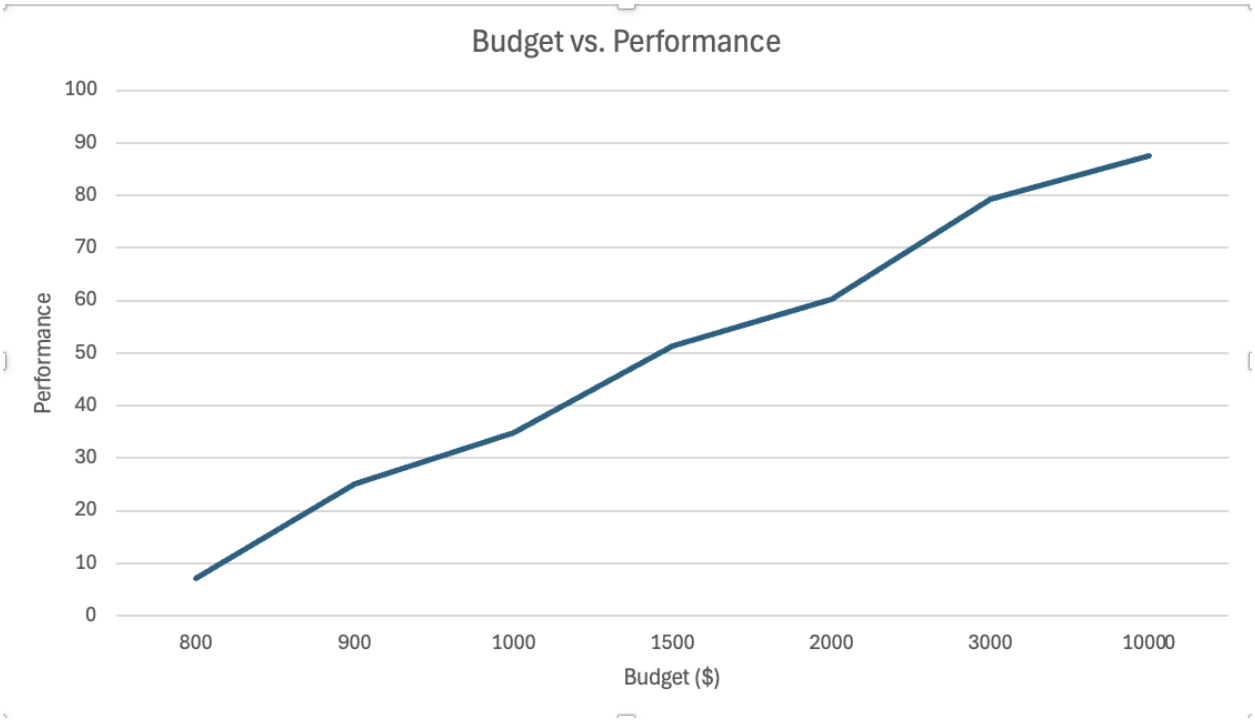
```
Optimal solution found (tolerance 1.00e-04)
Best objective 8.751600000000e+01, best bound 8.751600000000e+01, gap 0.0000%

Selected GPU:
G_10: Model=Asus PRIME GeForce RTX 5080, FPS=150, Color=Black, Price=999.99

Selected CPU:
C_5: Model=Ryzen 7 9800X3D, Base Frequency=4.7, Cores/Threads=0.5, Price=480, Brand=AMD

Selected motherboard:
M_17: Model=MSI MPG B550 GAMING PLUS, Memory max=128, Color=Black, Price=89.99, Brand=AMD

Selected RAM sticks:
R_13: Model=Corsair Vengeance 128 GB, DDR=5, Price=352.495, Number of Sticks=4.0

Selected Storage:
S_1: Model=Samsung 990 Pro 2 TB M.2-2280 PCIe 4.0 X4 NVME Solid State Drive, Storage Size (in GB)=2000, Price=189.99, Type=M.2SSD
S_6: Model=Samsung 870 Evo 1TB, Storage Size (in GB)=1000, Price=99.99, Type=2.5SSD
S_12: Model=Seagate BarraCuda 2TB, Storage Size (in GB)=2000, Price=65.99, Type=HDD

Selected Case:
E_12: Model=Phanteks Enthoo Pro, Size=69.1, Color=Black, Price=142.99

Selected Power Supply:
P_13: Model=Super Flower Leadex Titanium, Wattage=2800, Color=Black, Price=799.9

Selected Cooling unit:
L_3: Model=Noctua NH-D15 chromax.black, CPU temp over 21C=55.2, Color=Black, Price=153.08, Cooling Type=Air cooling

Selected Fan(s):
F_11: Model=ARCTIC S12038-8K, Rotations per minute (rpm)=8000, Color=Black, Price=15.99, Number of Fans=8.0

Total Cost:
4559.82
```

Since our model is a mixed-integer program, typical sensitivity measures such as dual variables, shadow prices, and reduced costs do not apply. Instead, we explored how the model solution changes as the budget, performance coefficients, and other user preferences change.

Budget Sensitivity Analysis

Since the user's budget is one of the most influential parameters, we tested our model using a range of different budgets while keeping other preferences the same (Color: Black, RGB: No). Below are figures showing the results for each budget tested:

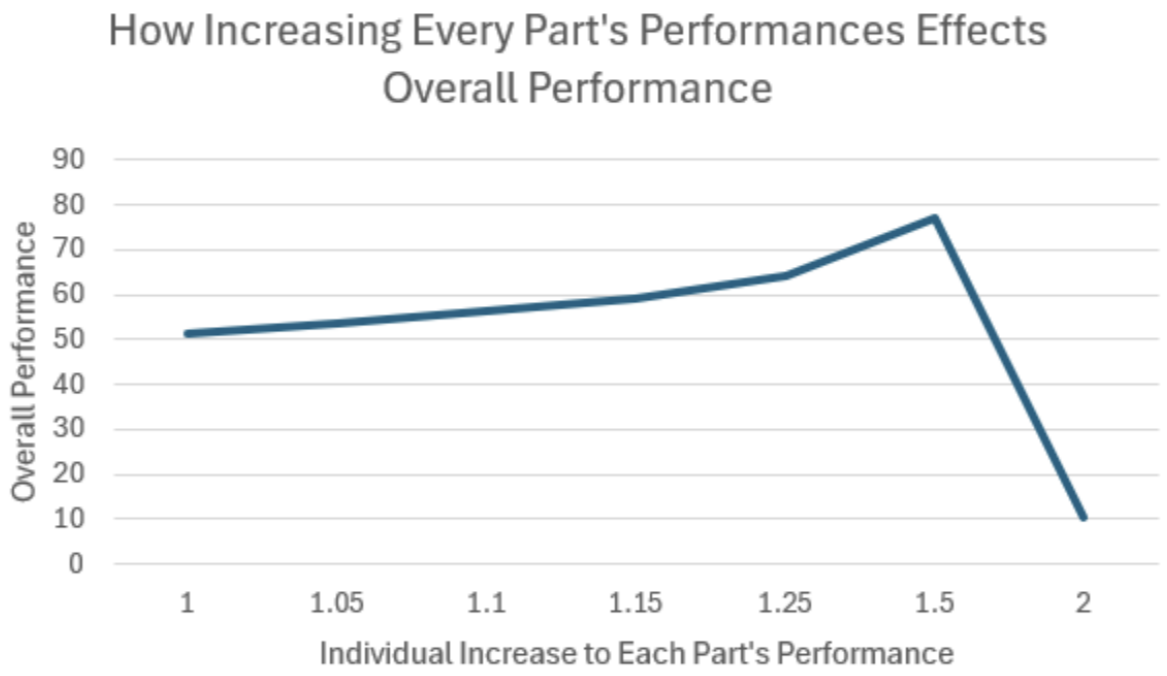
Budget ($)	Performance	Architecture	GPU
800	7.14	Intel	Sapphire PULSE Radeon RX 9060 XT
900	25.05	AMD	Sapphire PULSE Radeon RX 9060 XT
1000	34.87	AMD	Sapphire PULSE Radeon RX 9060 XT
1500	51.23	AMD	Sapphire PULSE Radeon RX 9060 XT
2000	60.23	AMD	Sapphire PULSE Radeon RX 9060 XT
3000	79.28	AMD	Sapphire PULSE Radeon RX 9060 XT
10000	87.52	AMD	Asus PRIME GeForce RTX 5080



As expected, the performance of the PC increased as budget increased. The relationship is relatively linear. Something unexpected was that up until an extremely high budget was reached, the GPU selected for each budget was the same ("Sapphire PULSE Radeon RX 9060 XT"). This indicates that this specific GPU reaches a very optimal performance value at a relatively low cost. Another observation is that the CPU architecture chosen for every budget except the lowest one was AMD, indicating that AMD is the overall choice for most builds, unless you are on an extremely tight budget.

Performance Weight Perturbation

We performed an analysis on increasing the performance score of every part by a fixed percentage to determine if it had an impact on the parts selected by the model.



As expected, the overall performance score goes up as the individual performances increase, but regardless of the budget limit, the part choices remain the same, and the same amount of money is spent each time. Here in the graph the budget is $1500, the color preference is black, and the user has no RGB preference.

Performance weight increase in GPU (%)	Overall Performance
1	72.47
1.2	72.9
1.5	73.55
2	74.6
Performance weight increase in Case	**Overall Performance**
1	72.47
1.2	72.62
1.5	72.86
2	73.24

We then performed a test to check if the importance of each part (their weight in the objective function) has the desired effect on the overall performance. Here the weight of the GPU (arguably the most important part) is being increased along with the weight of the Case (one of the least important). In the graph when the weight of each is increased, the GPU's increases a good bit faster, which is what we want to see and makes sense since its weight on the objective is higher.

Impact of Color on Performance

When holding the budget at a constant value of $4000 and setting RGB preference as any, the performance value for each color can be analyzed. When setting the color to pink the optimal value reported a performance of 68.22 and the total price was $3988.74. When setting the color to black the optimal value reported a performance of 87.58 and the total price was $3953.82. When setting the color to white the optimal value reported a performance of 79.71 and the total price was $3983.97. When setting the color to grey the optimal value reported a performance of 61.82 and the total price was $3753.16. When setting the color to rainbow the optimal value reported a performance of 69.18 and the total price was $3985.79. Ultimately, color constraints appear to significantly influence what components the optimizer selects. The differences in performance between colors comes almost entirely from the GPU and CPU/motherboard performance. This suggests that the black color option has the broadest or best component availability. The grey option appears to severely constrain the component selection, forcing trade-offs that harm performance.

Appendix

Video Link

https://youtu.be/9SyCkPg4j1w

Variable Encoding Spreadsheet

https://docs.google.com/spreadsheets/d/1PqUvuOeDYz3w5bLp5Ggmt4DBSJdclEQtLnNM-4WLCi0/edit?usp=sharing