# Hospital Cost Allocation Between Townships

Isaac Clark

May 5, 2025

## Introduction

Lets introduce a hypothetical scenario. Townships A, B, and C are all part of Salt Lake County. These three townships are in desperate need of new hospitals to serve their growing communities. Each township could build their own smaller hospitals to serve their community. If two townships work together they can build a medium hospital to serve their two communities, or if all three townships work together, they can build one large hospital to serve all three communities.

If each township builds a separate smaller hospital, they each shirk the brunt of the cost for the whole project, resulting in higher total cost for all the townships. If two townships work together, they'll need to build a medium sized hospital that can adequately serve both communities, which will be more expensive than the smaller hospital, but because the cost is shared between the two townships, each township will pay less than if they had built the smaller one on their own. Similarly, if all three townships work together to build an even more expensive large hospital that will be able to serve all three communities, the shared cost will be less than all other options.

It is our goal to figure out how much each township need to pay in order for all townships to cooperate.

## The Problem

As mentioned previously, we have three hypothetical townships A, B, and C all located in Salt Lake County, and they need a hospital. We will set up the problem as a cooperative game with the following values defined:

- $N = \{A, B, C\}$
- $C(S) =$ Cost for coalition $S$ to build the hospital

The characteristic function is defined as

$$v(S) = \sum_{i \in S} C(\{i\}) - C(S)$$

which will give us the value created, or what we'll refer to as the discount, by a coalition working together. For each coalition, we have an associated cost for building the hospital which are defined in the following table:

| Coalition $S$ | Cost $C(S)$ | Discount $v(S)$ |
|---|---|---|
| {A} | \$10M | \$0 |
| {B} | \$8M | \$0 |
| {C} | \$6M | \$0 |
| {A,B} | \$15M | \$3M |
| {A,C} | \$13M | \$3M |
| {B,C} | \$12M | \$2M |
| {A,B,C} | \$20M | \$4M |

Table 1: Coalition Costs

With this, we want to calculate the Shapley values and the Core allocations for each of the townships. We'll do this by using python. Running the calculations we get the following values:

| Township | Shapley Values | Core Allocations |
|---|---|---|
| A | 1.67 | 2 |
| B | 1.17 | 1 |
| C | 1.17 | 1 |

Table 2: Discount Allocation

These values represent the discount each township receives from the total cost due to cooperation. Using the allocations, the actual amount that each township pays is, Standalone Cost - Discount, as shown in the plot below:

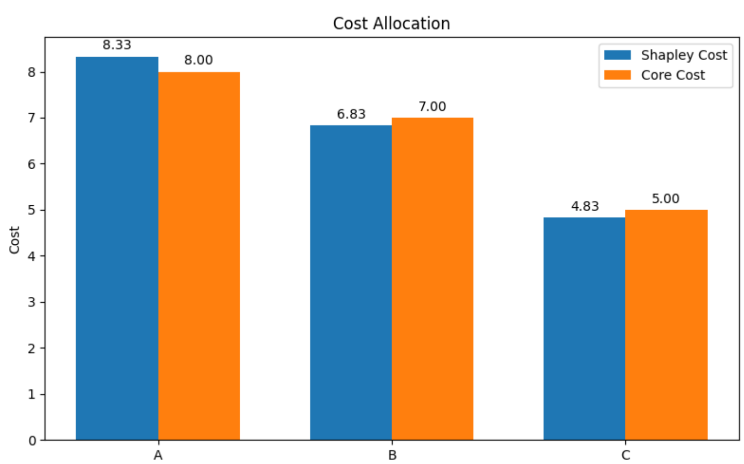


Figure 1: Cost Allocation

# Discussion

On first glance of the problem, it seems that the most fair allocation is the Shapley values, which is the purpose of the Shapley value. However, it is unstable. If we analyze the problem further, we see that the Shapley values are not in the core, and we can verify this with the following:

| Coalition $S$ | Shapley Value | Discount $v(S)$ | Check |
|---|---|---|---|
| {A} | 1.67 | 0 | $1.67 > 0$ |
| {B} | 1.17 | 0 | $1.17 > 0$ |
| {C} | 1.17 | 0 | $1.17 > 0$ |
| {A,B} | 2.84 | 3 | $2.84 < 3$ |
| {A,C} | 2.84 | 3 | $2.84 < 3$ |
| {B,C} | 2.34 | 2 | $2.34 > 2$ |

Table 3: Stability Verification

As we can see, coalitions {A,B} and {A,C} fail the check and are better deals for those respective coalitions than the Shapley values. If the Shapley values are chosen to allocate cost, then that might cause A and B or A and C to break off and no longer cooperate all together.

Because township A is footing most of the bill, they must be appeased with the greatest discount to maintain cooperation. However, Townships B and C are still getting a better deal then if they broke off and worked together or built the hospital on their own.

This allows us to conclude that the most optimal and cost effective choice for all townships if for them to all work together to build one large hospital that is capable of servicing all three communities and using the Core allocation to determine the discounts and cost for each township, which is shown in figure 1.

# Conclusion

While only a hypothetical example of the interactions between entities trying to complete a goal, This analysis shows that core allocations provide the necessary discounts for cooperation, while Shapley values may not ensure stability over fairness.

The resulting cooperative game used to model this hypothetical scenario has applications to a wide variety of scenarios big and small and has been used for decades. The ability to quantify problems of negotiation and cooperation mathematically is incredibly powerful.

As game theory is still a relatively young discipline, there is still much yet to be uncovered about the field of study and much more research should be done to even more better understand the powerful applications that game theory can provide to us as the world grows larger and more complex.

## Appendix: Python Code

```
import matplotlib.pyplot as plt
from scipy.optimize import linprog
import itertools
import math

players = ['A', 'B', 'C']
costs = {
    frozenset(['A']): 10,
    frozenset(['B']): 8,
    frozenset(['C']): 6,
    frozenset(['A', 'B']): 15,
    frozenset(['A', 'C']): 13,
    frozenset(['B', 'C']): 12,
    frozenset(['A', 'B', 'C']): 20
}

def value(coalition):
    if not coalition:
        return 0
    standalone_cost = sum(costs[frozenset([p])] for p in coalition)
    shared_cost = costs[frozenset(coalition)]
    return standalone_cost - shared_cost

def shapley_value(players, value):
    phi = dict.fromkeys(players, 0.0)

    for player in players:
        for coalition in itertools.chain.from_iterable(itertools.
            combinations([p for p in players if p != player], r) for r
            in range(len(players))):
             S = set(coalition)
             S_player = S | {player}
             marginal_contribution = value(S_player) - value(S)
             weight = (math.factorial(len(S)) * math.factorial(len(
                 players) - len(S) - 1)) / math.factorial(len(players))
             phi[player] += weight * marginal_contribution

    return phi

def core(players, value):
    coalitions = list(itertools.chain.from_iterable(itertools.
        combinations(players, r) for r in range(1, len(players))))
    A = []
    B = []

    for S in coalitions:
        A.append([-x for x in [1 if p in S else 0 for p in players]])
        B.append(-value(set(S)))
```

```
    A_eq = [[1] * len(players)]
    B_eq = [value(set(players))]

    return linprog(c=[0]*len(players), A_ub=A, b_ub=B, A_eq=A_eq, b_eq
       =B_eq, bounds=[(None, None)] * len(players))

shapley = shapley_value(players, lambda S: value(set(S)))
core_result = core(players, lambda S: value(set(S)))

print(shapley)
print(dict(zip(players, core_result.x)))

standalone_costs = {'A': 10, 'B': 8, 'C': 6}
total_joint_cost = 20

shapley_discount = {'A': 1.67, 'B': 1.17, 'C': 1.17}
core_discount = {'A': 2.0, 'B': 1.0, 'C': 1.0}

shapley_payments = {p: standalone_costs[p] - shapley_discount[p] for p
    in players}
core_payments = {p: standalone_costs[p] - core_discount[p] for p in
   players}

fig, ax = plt.subplots(figsize=(10, 6))

bar1 = ax.bar([i - 0.175 for i in range(len(players))],
   shapley_payments.values(), 0.35, label='Shapley Cost')
bar2 = ax.bar([i + 0.175 for i in range(len(players))], core_payments.
   values(), 0.35, label='Core Cost')

ax.set_title('Cost Allocation')
ax.set_ylabel('Cost')
ax.set_xticks(list(range(len(players))))
ax.set_xticklabels(players)
ax.legend()

for bar in bar1 + bar2:
    height = bar.get_height()
    ax.annotate(f'{height:.2f}',
        xy=(bar.get_x() + bar.get_width() / 2, height),
        xytext=(0, 3),
        textcoords="offset points",
        ha='center', va='bottom')

plt.tight_layout()
plt.show()
```