Greedy Algorithms
CSCI 432

Greedy Change Making (USD)

Greedy decision: Select the highest denomination coin that evenly subtracts from the remaining balance (i.e. max quarters + max dimes + max nickels + max pennies).

Proof of optimality:

Greedy Change Making (USD)

<u>Greedy decision:</u> Select the highest denomination coin that evenly subtracts from the remaining balance (i.e. max quarters + max dimes + max nickels + max pennies).

<u>Proof of optimality:</u> Let $V =$ \$ value. Let q, d, n, p be the number of quarters, dimes, nickels, and pennies used.

Greedy Change Making (USD)

Greedy decision: Select the highest denomination coin that evenly subtracts from the remaining balance (i.e. max quarters + max dimes + max nickels + max pennies).

Proof of optimality: Let $V =$ \$ value. Let q, d, n, p be the number of quarters, dimes, nickels, and pennies used.

Let $T_{ALG} = q_{ALG} + d_{ALG} + n_{ALG} + p_{ALG}$ be the algorithm's solution and $T_{OPT} = q_{OPT} + d_{OPT} + n_{OPT} + p_{OPT}$ be an optimal solution.

Greedy Change Making (USD)

Greedy decision: Select the highest denomination coin that evenly subtracts from the remaining balance (i.e. max quarters + max dimes + max nickels + max pennies).

Proof of optimality: Let $V =$ \$ value. Let q, d, n, p be the number of quarters, dimes, nickels, and pennies used.

Let $T_{ALG} = q_{ALG} + d_{ALG} + n_{ALG} + p_{ALG}$ be the algorithm's solution and $T_{OPT} = q_{OPT} + d_{OPT} + n_{OPT} + p_{OPT}$ be an optimal solution.

What can we say about q_{ALG} vs q_{OPT}?

Greedy Change Making (USD)

Greedy decision: Select the highest denomination coin that evenly subtracts from the remaining balance (i.e. max quarters + max dimes + max nickels + max pennies).

Proof of optimality: Let $V =$ \$ value. Let q, d, n, p be the number of quarters, dimes, nickels, and pennies used.

Let $T_{ALG} = q_{ALG} + d_{ALG} + n_{ALG} + p_{ALG}$ be the algorithm's solution and $T_{OPT} = q_{OPT} + d_{OPT} + n_{OPT} + p_{OPT}$ be an optimal solution.

$q_{OPT} \leq q_{ALG}$ since q_{ALG} is the largest value such that $V - 25\, q_{ALG} < 25$

Greedy Change Making (USD)

<u>Greedy decision:</u> Select the highest denomination coin that evenly subtracts from the remaining balance (i.e. max quarters + max dimes + max nickels + max pennies).

<u>Proof of optimality:</u> Let $V =$ \$ value. Let q, d, n, p be the number of quarters, dimes, nickels, and pennies used.

Let $T_{ALG} = q_{ALG} + d_{ALG} + n_{ALG} + p_{ALG}$ be the algorithm's solution and $T_{OPT} = q_{OPT} + d_{OPT} + n_{OPT} + p_{OPT}$ be an optimal solution.

$q_{OPT} \leq q_{ALG}$ since q_{ALG} is the largest value such that $V - 25\, q_{ALG} < 25$

If $q_{OPT} < q_{ALG}, V - 25\, q_{OPT} \geq 25$.

Why is this a problem?

Contradict the fact that T_{OPT} is optimal.

Greedy Change Making (USD)

<u>Greedy decision:</u> Select the highest denomination coin that evenly subtracts from the remaining balance (i.e. max quarters + max dimes + max nickels + max pennies).

<u>Proof of optimality:</u> Let $V = \$$ value. Let q, d, n, p be the number of quarters, dimes, nickels, and pennies used.

Let $T_{ALG} = q_{ALG} + d_{ALG} + n_{ALG} + p_{ALG}$ be the algorithm's solution and $T_{OPT} = q_{OPT} + d_{OPT} + n_{OPT} + p_{OPT}$ be an optimal solution.

$q_{OPT} \leq q_{ALG}$ since q_{ALG} is the largest value such that $V - 25\, q_{ALG} < 25$

If $q_{OPT} < q_{ALG}, V - 25\, q_{OPT} \geq 25.$

I.e. $V - 25\, q_{OPT} = 10\, d_{OPT} + 5\, n_{OPT} + p_{OPT}$

Greedy Change Making (USD)

Greedy decision: Select the highest denomination coin that evenly subtracts from the remaining balance (i.e. max quarters + max dimes + max nickels + max pennies).

Proof of optimality: Let $V = \$$ value. Let q, d, n, p be the number of quarters, dimes, nickels, and pennies used.

Let $T_{ALG} = q_{ALG} + d_{ALG} + n_{ALG} + p_{ALG}$ be the algorithm's solution and $T_{OPT} = q_{OPT} + d_{OPT} + n_{OPT} + p_{OPT}$ be an optimal solution.

$q_{OPT} \leq q_{ALG}$ since q_{ALG} is the largest value such that $V - 25\, q_{ALG} < 25$

If $q_{OPT} < q_{ALG}, V - 25\, q_{OPT} \geq 25$.

$$\begin{aligned} \text{I.e. } V - 25\, q_{OPT} &= 10\, d_{OPT} + 5\, n_{OPT} + p_{OPT} \\ &= 25 + 10(d_{OPT} - 2) + 5(n_{OPT} - 1) + p_{OPT} \end{aligned}$$

Greedy Change Making (USD)

<u>Greedy decision:</u> Select the highest denomination coin that evenly subtracts from the remaining balance (i.e. max quarters + max dimes + max nickels + max pennies).

<u>Proof of optimality:</u> Let $V =$ \$ value. Let q, d, n, p be the number of quarters, dimes, nickels, and pennies used.

Let $T_{ALG} = q_{ALG} + d_{ALG} + n_{ALG} + p_{ALG}$ be the algorithm's solution and $T_{OPT} = q_{OPT} + d_{OPT} + n_{OPT} + p_{OPT}$ be an optimal solution.

$q_{OPT} \leq q_{ALG}$ since q_{ALG} is the largest value such that $V - 25\, q_{ALG} < 25$

If $q_{OPT} < q_{ALG}$, $V - 25\, q_{OPT} \geq 25$.

I.e. $V - 25\, q_{OPT} = 10\, d_{OPT} + 5\, n_{OPT} + p_{OPT}$

$= 25 + 10(d_{OPT} - 2) + 5(n_{OPT} - 1) + p_{OPT}$

Technically we don't know if there are enough dimes and nickels for this, but if there are not, then it will require even more coins to account for the 25-cents.

Greedy Change Making (USD)

Greedy decision: Select the highest denomination coin that evenly subtracts from the remaining balance (i.e. max quarters + max dimes + max nickels + max pennies).

Proof of optimality: Let $V =$ \$ value. Let q, d, n, p be the number of quarters, dimes, nickels, and pennies used.

Let $T_{ALG} = q_{ALG} + d_{ALG} + n_{ALG} + p_{ALG}$ be the algorithm's solution and $T_{OPT} = q_{OPT} + d_{OPT} + n_{OPT} + p_{OPT}$ be an optimal solution.

$q_{OPT} \leq q_{ALG}$ since q_{ALG} is the largest value such that $V - 25\, q_{ALG} < 25$

If $q_{OPT} < q_{ALG}, V - 25\, q_{OPT} \geq 25.$

$$\begin{aligned} \text{I.e. } V - 25\, q_{OPT} &= 10\, d_{OPT} + 5\, n_{OPT} + p_{OPT} \\ &= 25 + 10(d_{OPT} - 2) + 5(n_{OPT} - 1) + p_{OPT} \\ \Rightarrow V &= 25(q_{OPT} + 1) + 10(d_{OPT} - 2) + 5(n_{OPT} - 1) + p_{OPT} \end{aligned}$$

Greedy Change Making (USD)

Greedy decision: Select the highest denomination coin that evenly subtracts from the remaining balance (i.e. max quarters + max dimes + max nickels + max pennies).

Proof of optimality: Let $V =$ \$ value. Let q, d, n, p be the number of quarters, dimes, nickels, and pennies used.

Let $T_{ALG} = q_{ALG} + d_{ALG} + n_{ALG} + p_{ALG}$ be the algorithm's solution and $T_{OPT} = q_{OPT} + d_{OPT} + n_{OPT} + p_{OPT}$ be an optimal solution.

$q_{OPT} \leq q_{ALG}$ since q_{ALG} is the largest value such that $V - 25\, q_{ALG} < 25$

If $q_{OPT} < q_{ALG}, V - 25\, q_{OPT} \geq 25.$

$$
\begin{aligned}
\text{I.e. } V - 25\, q_{OPT} &= 10\, d_{OPT} + 5\, n_{OPT} + p_{OPT} \\
&= 25 + 10(d_{OPT} - 2) + 5(n_{OPT} - 1) + p_{OPT}
\end{aligned}
$$

$$\Rightarrow V = 25(q_{OPT} + 1) + 10(d_{OPT} - 2) + 5(n_{OPT} - 1) + p_{OPT}$$

$$\Rightarrow \text{Num coins} = q_{OPT} + d_{OPT} + n_{OPT} + p_{OPT} - 2$$

Greedy Change Making (USD)

Greedy decision: Select the highest denomination coin that evenly subtracts from the remaining balance (i.e. max quarters + max dimes + max nickels + max pennies).

Proof of optimality: Let $V =$ \$ value. Let q, d, n, p be the number of quarters, dimes, nickels, and pennies used.

Let $T_{ALG} = q_{ALG} + d_{ALG} + n_{ALG} + p_{ALG}$ be the algorithm's solution and $T_{OPT} = q_{OPT} + d_{OPT} + n_{OPT} + p_{OPT}$ be an optimal solution.

$q_{OPT} \leq q_{ALG}$ since q_{ALG} is the largest value such that $V - 25\, q_{ALG} < 25$

If $q_{OPT} < q_{ALG}, V - 25\, q_{OPT} \geq 25.$

I.e. $V - 25\, q_{OPT} = 10\, d_{OPT} + 5\, n_{OPT} + p_{OPT}$

$= 25 + 10(d_{OPT} - 2) + 5(n_{OPT} - 1) + p_{OPT}$

$\Rightarrow V = 25(q_{OPT} + 1) + 10(d_{OPT} - 2) + 5(n_{OPT} - 1) + p_{OPT}$

$\Rightarrow$ Num coins $= q_{OPT} + d_{OPT} + n_{OPT} + p_{OPT} - 2 < T_{OPT}$. Contradiction.

Thus, $q_{OPT} = q_{ALG}$.

Greedy Change Making (USD)

<u>Greedy decision:</u> Select the highest denomination coin that evenly subtracts from the remaining balance (i.e. max quarters + max dimes + max nickels + max pennies).

<u>Proof of optimality:</u> Let $V =$ \$ value. Let q, d, n, p be the number of quarters, dimes, nickels, and pennies used.

Let $T_{ALG} = q_{ALG} + d_{ALG} + n_{ALG} + p_{ALG}$ be the algorithm's solution and $T_{OPT} = q_{OPT} + d_{OPT} + n_{OPT} + p_{OPT}$ be an optimal solution.

$q_{OPT} \leq q_{ALG}$ since q_{ALG} is the largest value such that $V - 25\, q_{ALG} < 25$

If $q_{OPT} < q_{ALG}$, $V - 25\, q_{OPT} \geq 25$.

I.e. $V - 25\, q_{OPT} = 10\, d_{OPT} + 5\, n_{OPT} + p_{OPT}$

$= 25 + 10(d_{OPT} - 2) + 5(n_{OPT} - 1) + p_{OPT}$

$\Rightarrow V = 25(q_{OPT} + 1) + 10(d_{OPT} - 2) + 5(n_{OPT} - 1) + p_{OPT}$

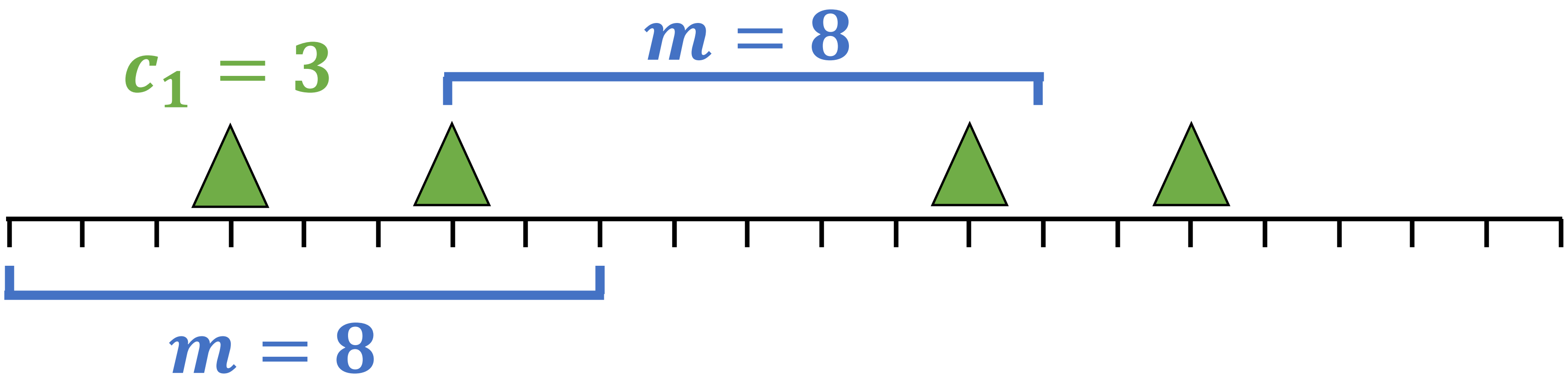
$\Rightarrow$ Num coins $= q_{OPT} + d_{OPT} + n_{OPT} + p_{OPT} - 2 < T_{OPT}$. Contradiction.

Thus, $q_{OPT} = q_{ALG}$.

Repeat argument to show that $d_{OPT} = d_{ALG}$, $n_{OPT} = n_{ALG}$, and $p_{OPT} = p_{ALG}$. Thus, $T_{OPT} = T_{ALG}$ and T_{ALG} is optimal.

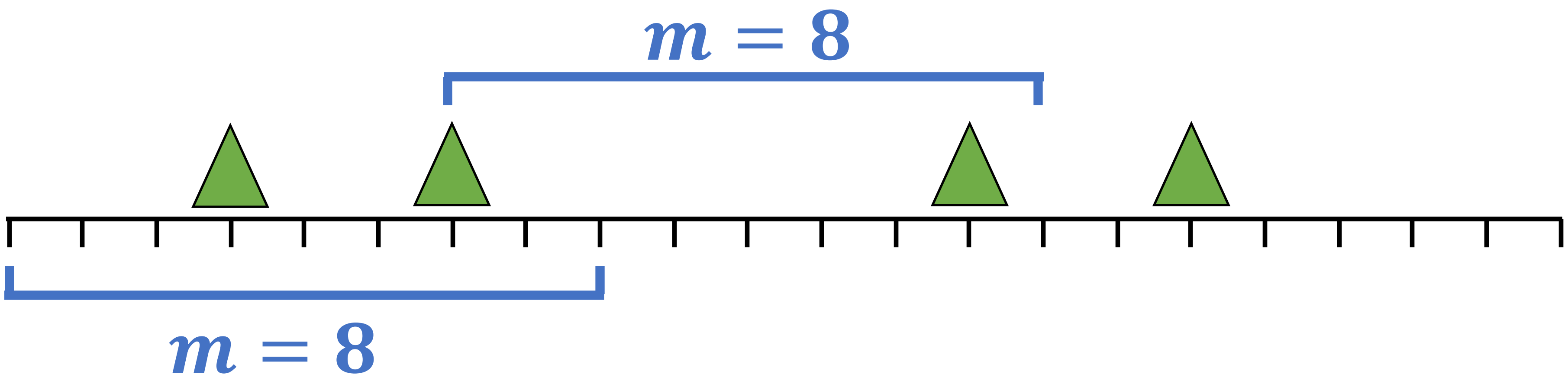
Problem Statement

Suppose you are going on a long backpacking trip on some trail. You can hike up to m miles a day. There are designated campsites along the trail at miles $(c_1, c_2, \ldots, c_n)$. You need to stay in a designated campsite each night. Your goal is to hike the whole trail in as few days as possible.



Problem Statement

Suppose you are going on a long backpacking trip on some trail. You can hike up to m miles a day. There are designated campsites along the trail at miles $(c_1, c_2, \ldots, c_n)$. You need to stay in a designated campsite each night. Your goal is to hike the whole trail in as few days as possible.

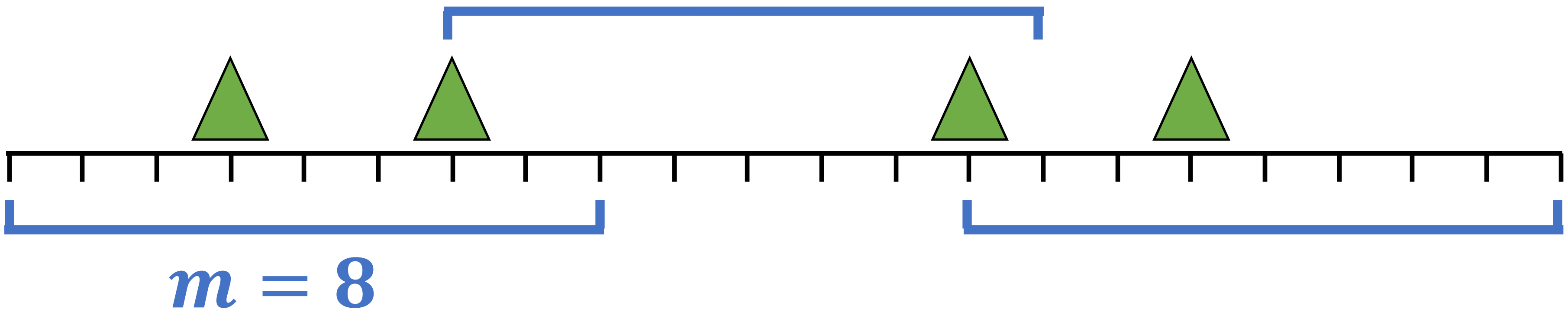


Come up with a greedy algorithm that will provide a valid solution (i.e. don't worry about it being optimal right now).

Greedy Algorithm

Greedy Campsite Selection (GCS): Camp at the last reachable (i.e. $< m$) campsite each day.

Is GCS valid?



Greedy Algorithm

```
campsite_selection(campsites C, maxDistance m)

                ??
```

Greedy Algorithm

```
campsite_selection(campsites C, maxDistance m)
  C.sort() //if needed
  lastSpot = 0
  for i = 1 to C.length
    if lastSpot + m < C[i]
      selected.add(C[i - 1])
      lastSpot = C[i - 1]
  return selected
```

Greedy Algorithm

```
campsite_selection(campsites C, maxDistance m)
  C.sort() //if needed
  lastSpot = 0
  for i = 1 to C.length
    if lastSpot + m < C[i]
      selected.add(C[i - 1])
      lastSpot = C[i - 1]
  return selected
```

Running time?

Greedy Algorithm

```
campsite_selection(campsites C, maxDistance m)
    C.sort() //if needed
    lastSpot = 0
    for i = 1 to C.length
        if lastSpot + m < C[i]
            selected.add(C[i - 1])
            lastSpot = C[i - 1]
    return selected
```

Running time?

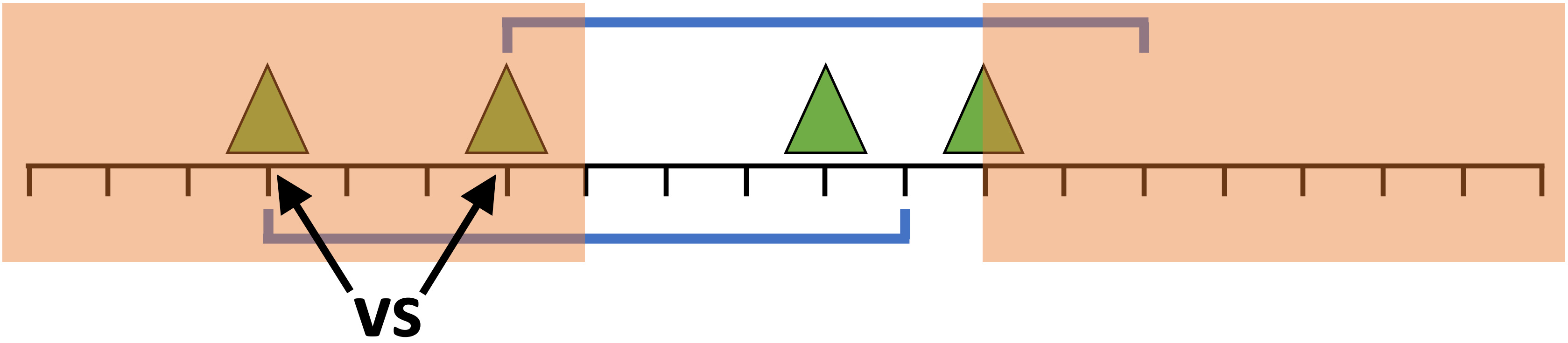
$n = |C|$

$O(n \log n)$ - If sorting needed

$O(n)$ - If sorting not needed

Greedy Algorithm

Greedy Campsite Selection (GCS): Camp at the last reachable (i.e. $< m$) campsite each day.



Will this ever not be optimal? Does it ever make sense to end the day early, to help with campsite spacing on a future day?

Theorem: Camping at the last reachable campsite each day is optimal.

Proof:

Theorem: Camping at the last reachable campsite each day is optimal.

Proof:

Plan?

Greedy Algorithm

Theorem: Camping at the last reachable campsite each day is optimal.

Proof:

Prove this by arguing that you can swap S_{ALG} campsites into S_{OPT} and keep S_{OPT} both valid and optimal.

Theorem: Camping at the last reachable campsite each day is optimal.

Proof: Let $S_{ALG} = (c_{a_1}, \dots, c_{a_k})$ be the campsites chosen by GCS and $S_{OPT} = (c_{o_1}, \dots, c_{o_t})$ be an optimal selection.

Theorem: Camping at the last reachable campsite each day is optimal.

Proof: Let $S_{ALG} = (c_{a_1}, \dots, c_{a_k})$ be the campsites chosen by GCS and $S_{OPT} = (c_{o_1}, \dots, c_{o_t})$ be an optimal selection.

Suppose night i is the first night that S_{ALG} and S_{OPT} do not stop at the same site.

Theorem: Camping at the last reachable campsite each day is optimal.

Proof: Let $S_{ALG} = (c_{a_1}, \dots, c_{a_k})$ be the campsites chosen by GCS and $S_{OPT} = (c_{o_1}, \dots, c_{o_t})$ be an optimal selection.

Suppose night i is the first night that S_{ALG} and S_{OPT} do not stop at the same site.

Consider $S'_{OPT} = S_{OPT} \setminus \{c_{o_i}\} \cup \{c_{a_i}\}$.

Theorem: Camping at the last reachable campsite each day is optimal.

Proof: Let $S_{ALG} = (c_{a_1}, \dots, c_{a_k})$ be the campsites chosen by GCS and $S_{OPT} = (c_{o_1}, \dots, c_{o_t})$ be an optimal selection.

Suppose night i is the first night that S_{ALG} and S_{OPT} do not stop at the same site.

Consider $S'_{OPT} = S_{OPT} \setminus \{c_{o_i}\} \cup \{c_{a_i}\}$.

c_{a_i} will not require S_{OPT} to go farther than m on day i, since it worked in S_{ALG} and $c_{a_{i-1}} = c_{o_{i-1}}$.

Theorem: Camping at the last reachable campsite each day is optimal.

Proof: Let $S_{ALG} = (c_{a_1}, \dots, c_{a_k})$ be the campsites chosen by GCS and $S_{OPT} = (c_{o_1}, \dots, c_{o_t})$ be an optimal selection.

Suppose night i is the first night that S_{ALG} and S_{OPT} do not stop at the same site.

Consider $S'_{OPT} = S_{OPT} \setminus \{c_{o_i}\} \cup \{c_{a_i}\}$.

c_{a_i} will not require S_{OPT} to go farther than m on day i, since it worked in S_{ALG} and $c_{a_{i-1}} = c_{o_{i-1}}$. Likewise, c_{a_i} will also work for $c_{o_{i+1}}$ since $c_{o_i} \leq c_{a_i}$, so if $c_{o_{i+1}} - c_{o_i} \leq m$ then $c_{o_{i+1}} - c_{a_i} \leq m$ too.

Theorem: Camping at the last reachable campsite each day is optimal.

Proof: Let $S_{ALG} = (c_{a_1}, \dots, c_{a_k})$ be the campsites chosen by GCS and $S_{OPT} = (c_{o_1}, \dots, c_{o_t})$ be an optimal selection.

Suppose night i is the first night that S_{ALG} and S_{OPT} do not stop at the same site.

Consider $S'_{OPT} = S_{OPT} \setminus \{c_{o_i}\} \cup \{c_{a_i}\}$.

c_{a_i} will not require S_{OPT} to go farther than m on day i, since it worked in S_{ALG} and $c_{a_{i-1}} = c_{o_{i-1}}$. Likewise, c_{a_i} will also work for $c_{o_{i+1}}$ since $c_{o_i} \leq c_{a_i}$, so if $c_{o_{i+1}} - c_{o_i} \leq m$ then $c_{o_{i+1}} - c_{a_i} \leq m$ too.

So S'_{OPT} is valid (doesn't exceed m-distance restriction) and has the same number of sites as S_{OPT}, so it's optimal.

Theorem: Camping at the last reachable campsite each day is optimal.

Proof: Let $S_{ALG} = (c_{a_1}, \dots, c_{a_k})$ be the campsites chosen by GCS and $S_{OPT} = (c_{o_1}, \dots, c_{o_t})$ be an optimal selection.

Suppose night i is the first night that S_{ALG} and S_{OPT} do not stop at the same site.

Consider $S'_{OPT} = S_{OPT} \setminus \{c_{o_i}\} \cup \{c_{a_i}\}$.

c_{a_i} will not require S_{OPT} to go farther than m on day i, since it worked in S_{ALG} and $c_{a_{i-1}} = c_{o_{i-1}}$. Likewise, c_{a_i} will also work for $c_{o_{i+1}}$ since $c_{o_i} \leq c_{a_i}$, so if $c_{o_{i+1}} - c_{o_i} \leq m$ then $c_{o_{i+1}} - c_{a_i} \leq m$ too.

We can repeat this by replacing campsites in S_{OPT} with the corresponding site in S_{ALG} without exceeding the m-restriction or increasing the number of sites.

Theorem: Camping at the last reachable campsite each day is optimal.

Proof: Let $S_{ALG} = (c_{a_1}, \dots, c_{a_k})$ be the campsites chosen by GCS and $S_{OPT} = (c_{o_1}, \dots, c_{o_t})$ be an optimal selection.

Suppose night i is the first night that S_{ALG} and S_{OPT} do not stop at the same site.

Consider $S'_{OPT} = S_{OPT} \setminus \{c_{o_i}\} \cup \{c_{a_i}\}$.

c_{a_i} will not require S_{OPT} to go farther than m on day i, since it worked in S_{ALG} and $c_{a_{i-1}} = c_{o_{i-1}}$. Likewise, c_{a_i} will also work for $c_{o_{i+1}}$ since $c_{o_i} \le c_{a_i}$, so if $c_{o_{i+1}} - c_{o_i} \le m$ then $c_{o_{i+1}} - c_{a_i} \le m$ too.

We can repeat this by replacing campsites in S_{OPT} with the corresponding site in S_{ALG} without exceeding the m-restriction or increasing the number of sites.

Thus, S_{ALG} is optimal.