# Introduction à l'Intelligence Artificielle

## Cours 2 – mercredi 05 octobre 2022

Adrien Revault d'Allonnes

ara@up8.edu

Université Paris 8 – Vincennes à Saint-Denis

IIA – sept. à déc., 2022

# Résolution du problème

- Trouver une séquence d'actions pour mener de l'état initial à un état but
  - construction d'un **graphe d'états**
- Stratégies de résolution
  - méthodes **exhaustives** : parcours systématique de l'espace des états
    - méthode aléatoire
    - recherche en largeur
    - recherche en profondeur
  - méthodes **informées** ou **heuristiques** : parcours de l'espace des états guidé par une connaissance
    - utilisation d'une heuristique
    - algorithme glouton
    - algorithme A*

# Méthodes informées ou heuristiques

- *'A process that may solve a given problem, but offers no guarantee of doing so, is called a* ***heuristic*** *for that problem'*
  (Newel, Shaw & Simon, 1963)
- Améliorer la recherche en utilisant des connaissances du domaine
  - évaluer les états lors de la recherche d'un chemin vers la solution

$\Rightarrow$ **Fonction d'évaluation** $h$ des états pour en guider l'examen

- $h$ : États $\longrightarrow \mathbb{R}^+$
- si $h(e) = 0$, alors $e$ est un état but
- si $h(e) = +\infty$, alors $e$ est un état interdit ou sans successeurs
- $\forall e \in$ États, $h(e)$ donne une **estimation** de la distance de $e$ au but

- Remarques
  - il n'est généralement pas facile de déterminer une fonction $h$
  - $h$ doit s'appuyer sur des connaissances que l'on a sur le problème
  - on dit que $h$ est une **fonction heuristique**
- $h$ sert donc à guider la recherche du chemin vers la solution

# Méthode heuristique : un premier algorithme

- But : améliorer l'examen des états grâce à une fonction d'évaluation
- Algorithme glouton == Greedy algorithm
  - toujours faire le choix localement optimal
- Étant donné une **fonction heuristique** $h$ : États $\longrightarrow \mathbb{R}^+$
  - pour tout $e, h(e)$ donne une **estimation** de la distance de $e$ au but
  - idée : toujours essayer de s'avancer vers le but
    - toujours réduire sa distance au but en choisissant l'action qui nous en rapproche le plus
- **Algorithme glouton** : à chaque étape, choisir le minimum local
  - tant qu'il reste des états non visités
    - soit $e$ l'état courant et $E$ les états déjà visités
    - soit $S = \{e' \notin E, \text{tel que } e \longrightarrow e'\}$
    - choisir $e' \in S$ tel que $h(e') = \min\{h(s), s \in S\}$
    - si $h(e') = +\infty$, alors arrêt, sinon $e'$ devient l'état courant

# Exemple de recherche gloutonne

- Rendu de monnaie
  - atteindre la somme $d$ donnée, à l'aide de pièces de 1, 2 et 5 centimes
- Modélisation du problème
  - états : somme $x \in \mathbb{N}$, état initial $x_0 = 0$
  - actions : $+1, +2, +5$
  - but : atteindre la somme $d$
- Fonction heuristique : $h$ : États $\longrightarrow \mathbb{R}^+$
  - $h(x) = \begin{cases} d - x, & \text{si } x \leq d \\ +\infty & \text{sinon} \end{cases}$
- Algorithme glouton
  - mise en œuvre : choisir toujours la pièce la plus haute
- Exemple de résolution
  - rendre 13 centimes
  - $0 \longrightarrow 5 \longrightarrow 5+5 \longrightarrow 5+5+2 \longrightarrow 5+5+2+1$

# Méthode heuristique : l'algorithme glouton

- Limites
  - problèmes s'il faut commencer par s'éloigner du but pour l'atteindre
  - l'algorithme échoue sur certains problèmes
    - choisir un minimum local à chaque étape ne permet pas forcément d'atteindre le minimum global
  - par exemple, pour le rendu de monnaie
    - rendre $d = 6$ centimes avec des pièces de 2 et 5
- Il manque un mécanisme de « retour arrière » pour revenir explorer des choix mis de côté

# Méthode heuristique : l'algorithme A*

- Idées :
  - garder une trace des évaluations déjà rencontrées et les utiliser pour choisir son chemin
- L'algorithme A*
  - algorithme de recherche « meilleur d'abord »
  - exploration « par sauts » du graphe de recherche
- Principes
  - coût d'un état $e$ : $f(e) = g(e) + h(e)$
    - $g(e)$ : coût mesuré pour venir de l'état initial à l'état courant $e$
    - $h(e)$ : coût estimé pour aller de $e$ à l'état final (heuristique)
  - une heuristique $h$ est **admissible** si elle ne surestime jamais le coût pour atteindre le but

# Méthode heuristique l'algorithme A*

- Initialisation
  - soit
    - $e_0$ : l'état initial
    - $F = \varnothing$ : file d'attente des états à évaluer
    - $V = \varnothing$ : états évalués
  - init : $F \longleftarrow (e_0, f(e_0))$ (NB : on a $g(e_0) = 0$ donc $f(e_0) = h(e_0)$)
- tant que $e$ n'est pas terminal et que $F$ n'est pas vide
  - $e \longleftarrow$ PlusPetitCout$(F)$
  - enlever$(F, (e, f(e)))$
  - ajouter$(V, (e, f(e)))$
  - pour chaque successeur $s$ de $e$
    - calculer $f(s)$ et ajouter$(F, (s, f(s)))$
  - fin pour
- fin tant que
- si $e$ est terminal reconstituer le chemin de $e_0$ à $e$
  - succès
- sinon, si $F$ est vide
  - échec

# Méthode heuristique l'algorithme A*

- Terminaison : si le graphe des états est fini, alors A* s'arrête
  - la file d'états est vide, si et seulement s'il n'existe pas de chemin de l'état initial à la solution
- Exemple d'application : le taquin
  - passer de $e_0 = [283; 164; 705]$ à $e_T = [123; 804; 765]$
  - 0 représente la case vide
  - fonction d'évaluation
    - $g(e)$ : nombre de coups pour passer de $e_0$ à $e$
    - $h(e)$ : nombre de cases non nulles mal placées
- Exercice :
  - dérouler l'algorithme A* pour trouver la séquence de coups permettant d'aller de $e_0$ à $e_T$
  - construire le graphe des états explorés correspondant

$e_T =$

| | | |
|---|---|---|
| 1 | 2 | 3 |
| 8 | _ | 4 |
| 7 | 6 | 5 |

| | | |
|---|---|---|
| 2 | 8 | 3 |
| 1 | 6 | 4 |
| 7 | 0 | 5 |

$f(e) = g(e) + h(e)$

$e_T =$

| | | |
|---|---|---|
| 1 | 2 | 3 |
| 8 | _ | 4 |
| 7 | 6 | 5 |

$f(e) = g(e) + h(e)$

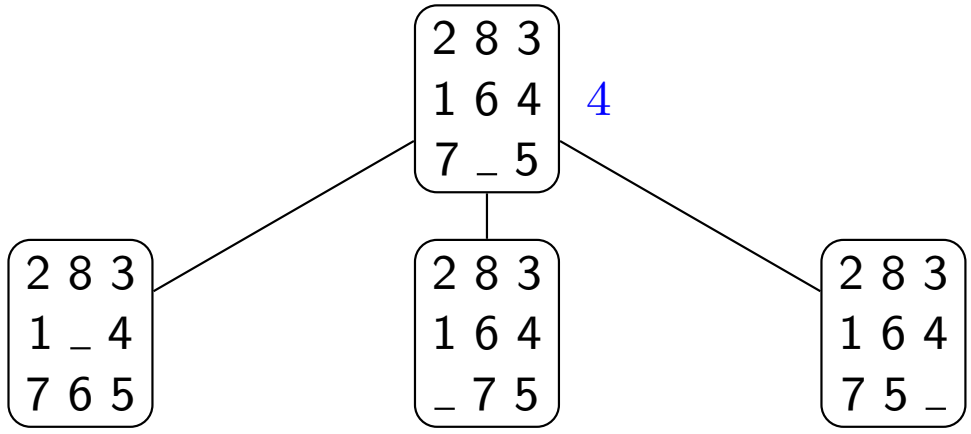
| | | |
|---|---|---|
| 2 | 8 | 3 |
| 1 | 6 | 4 |
| 7 | _ | 5 |

$4 = 0 + 4$

$e_T =$

| | | |
|---|---|---|
| 1 | 2 | 3 |
| 8 | _ | 4 |
| 7 | 6 | 5 |

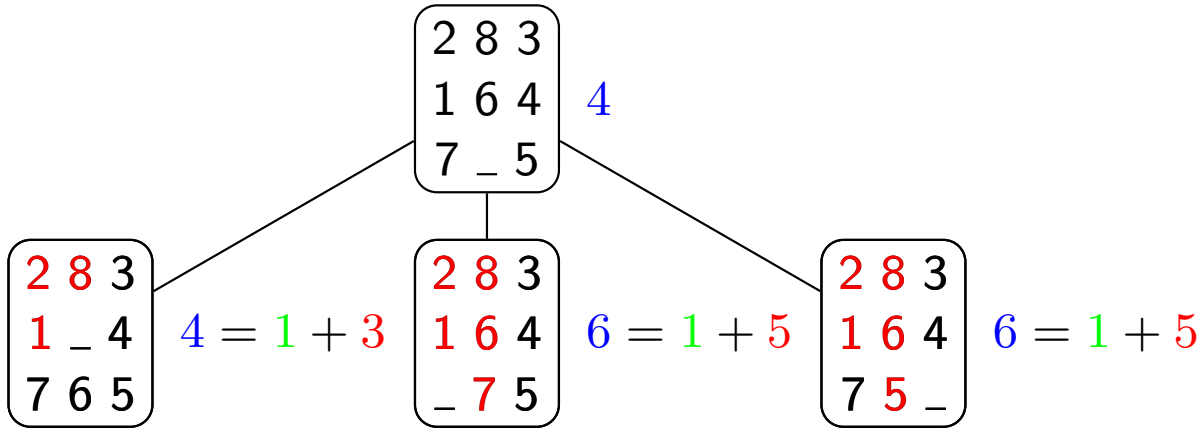
$f(e) = g(e) + h(e)$



$e_T =$

| | | |
|---|---|---|
| 1 | 2 | 3 |
| 8 | _ | 4 |
| 7 | 6 | 5 |

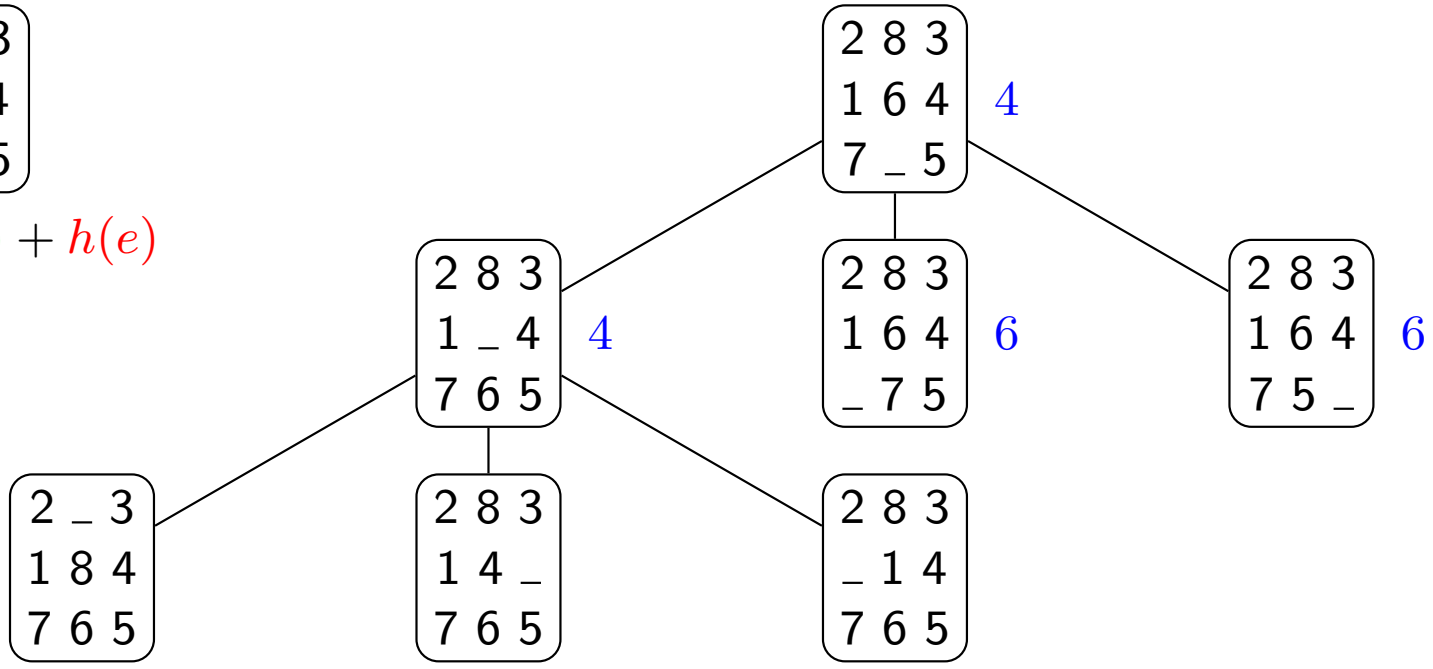
$f(e) = g(e) + h(e)$



$e_T =$

| | | |
|---|---|---|
| 1 | 2 | 3 |
| 8 | _ | 4 |
| 7 | 6 | 5 |

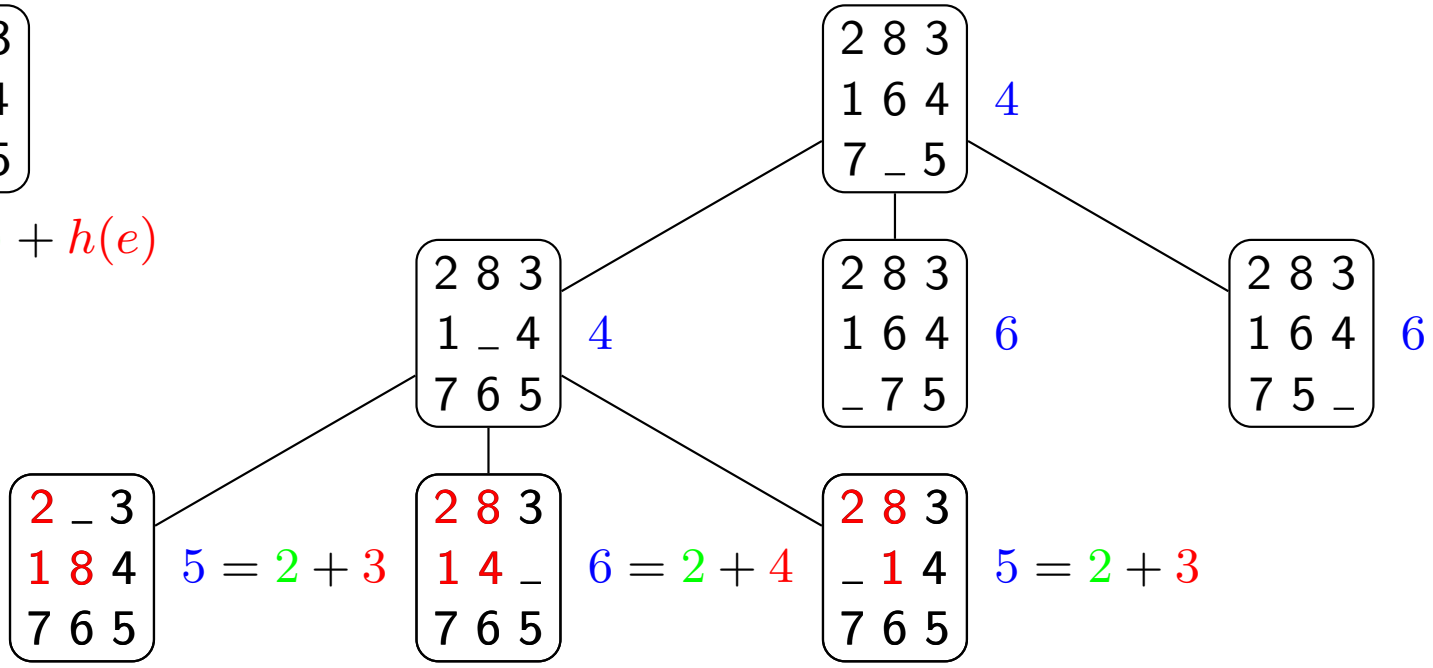
$f(e) = g(e) + h(e)$



$e_T =$

| | | |
|---|---|---|
| 1 | 2 | 3 |
| 8 | _ | 4 |
| 7 | 6 | 5 |

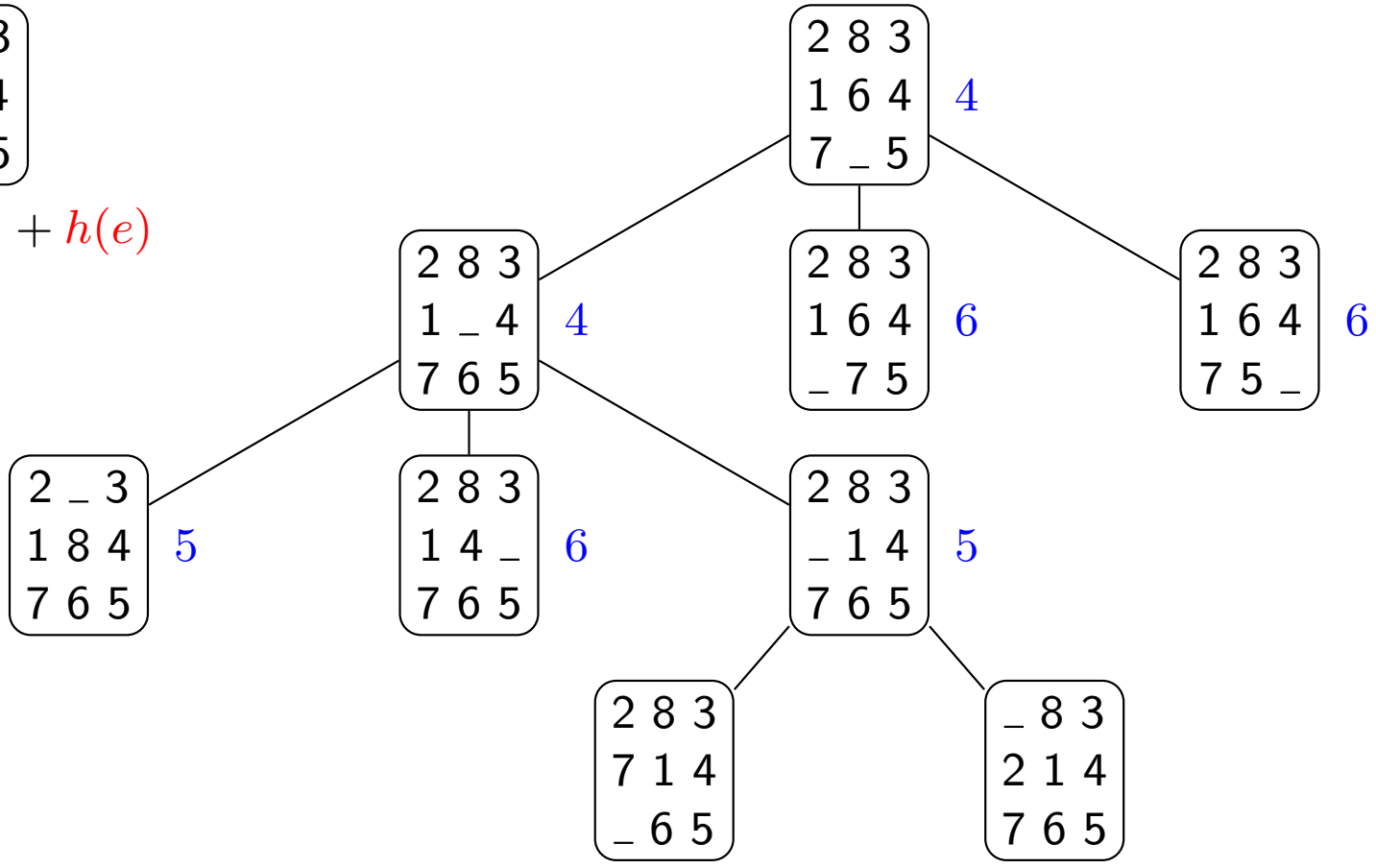
$f(e) = g(e) + h(e)$



$e_T =$

| | | |
|---|---|---|
| 1 | 2 | 3 |
| 8 | _ | 4 |
| 7 | 6 | 5 |

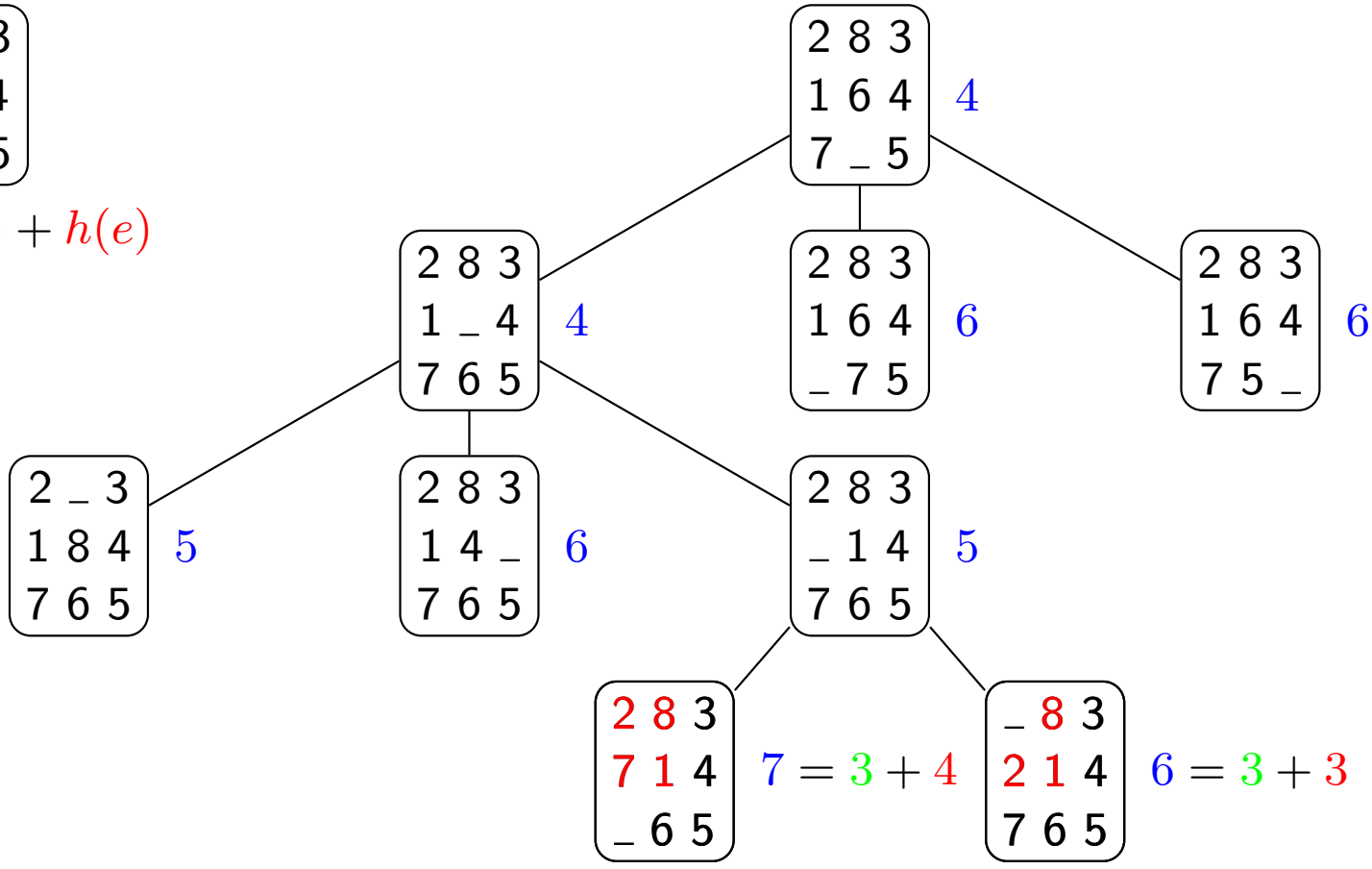
$f(e) = g(e) + h(e)$



$e_T =$

| | | |
|---|---|---|
| 1 | 2 | 3 |
| 8 | _ | 4 |
| 7 | 6 | 5 |

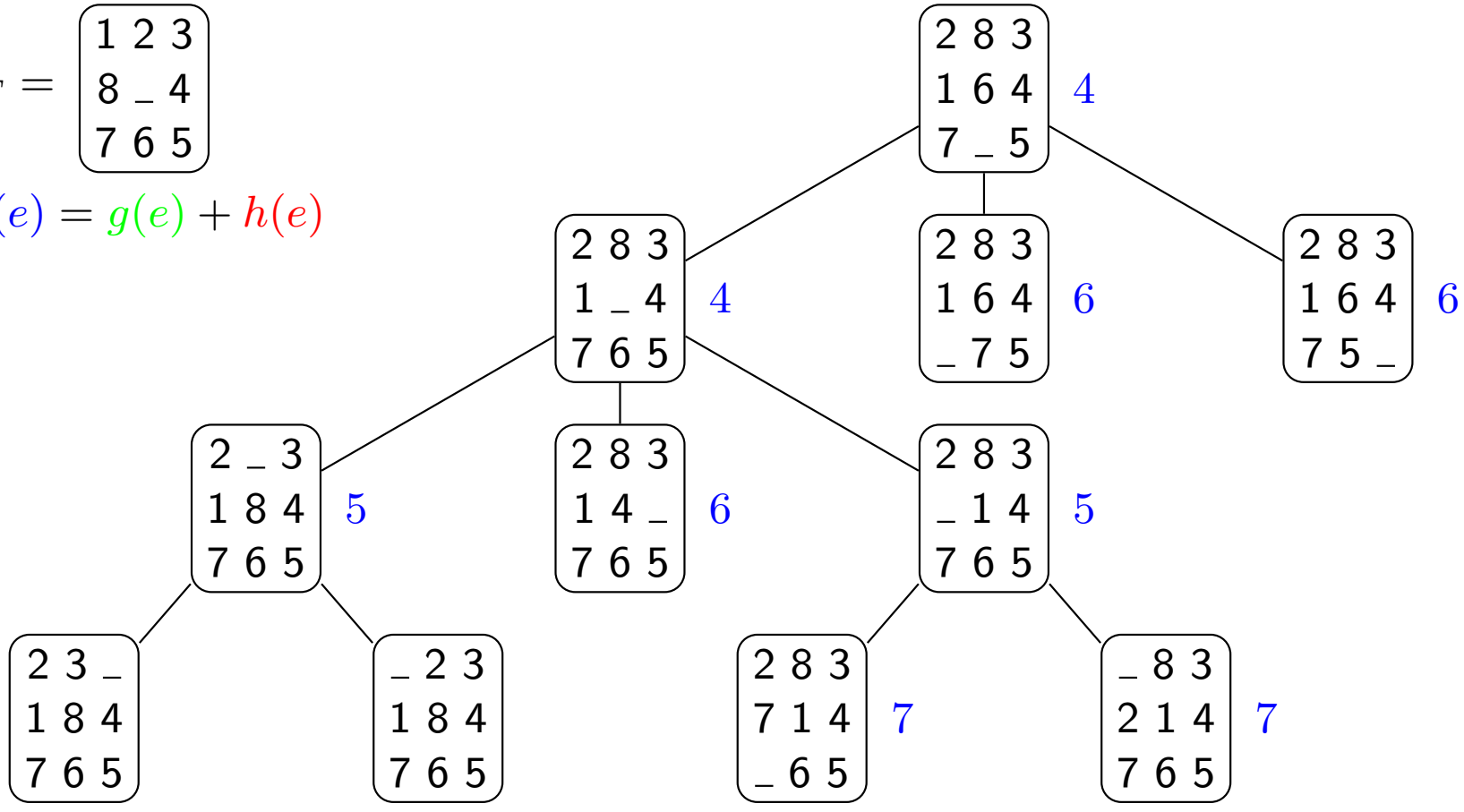
$f(e) = g(e) + h(e)$

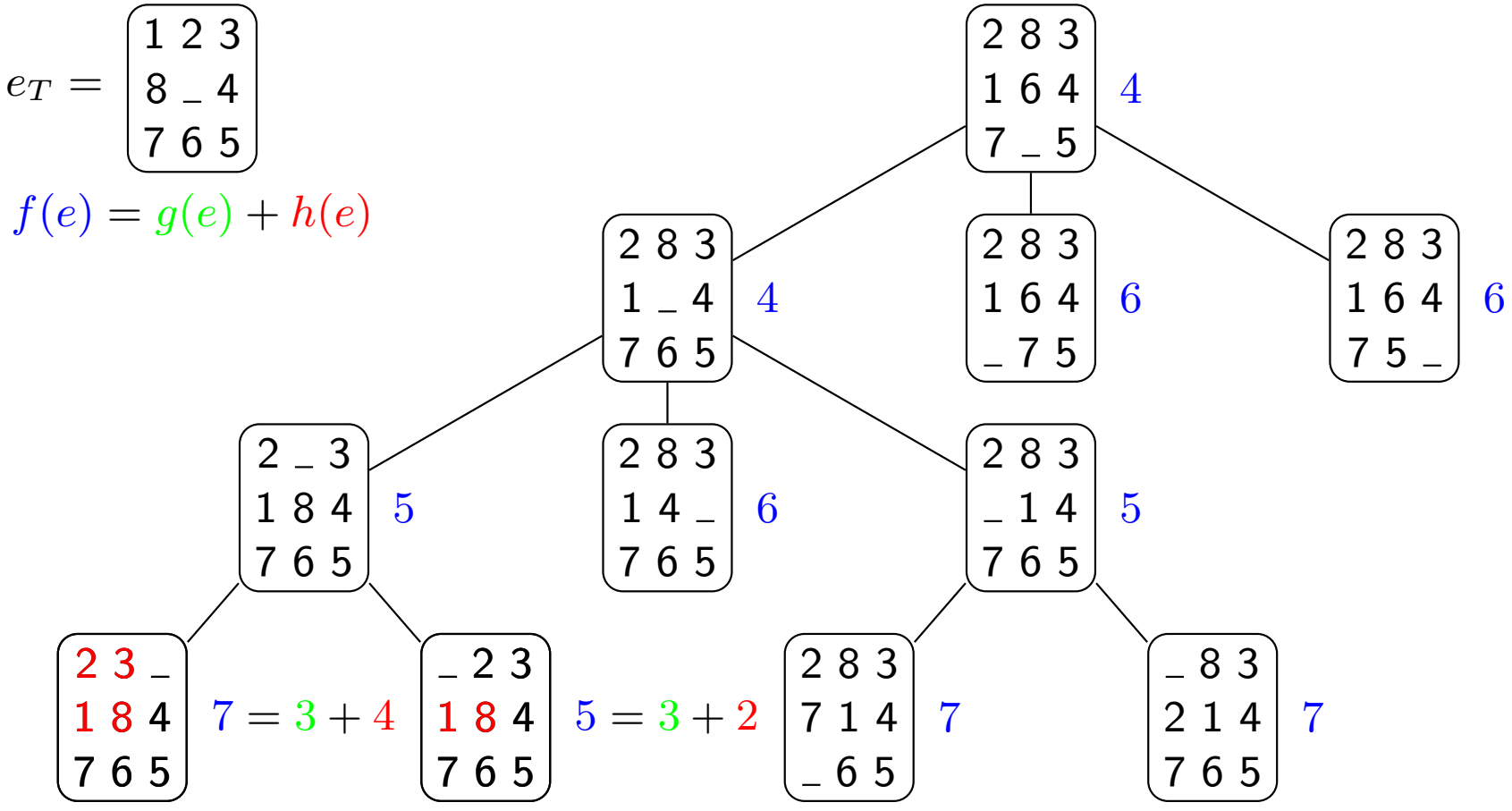


$e_T =$

| | | |
|---|---|---|
| 1 | 2 | 3 |
| 8 | _ | 4 |
| 7 | 6 | 5 |

$f(e) = g(e) + h(e)$

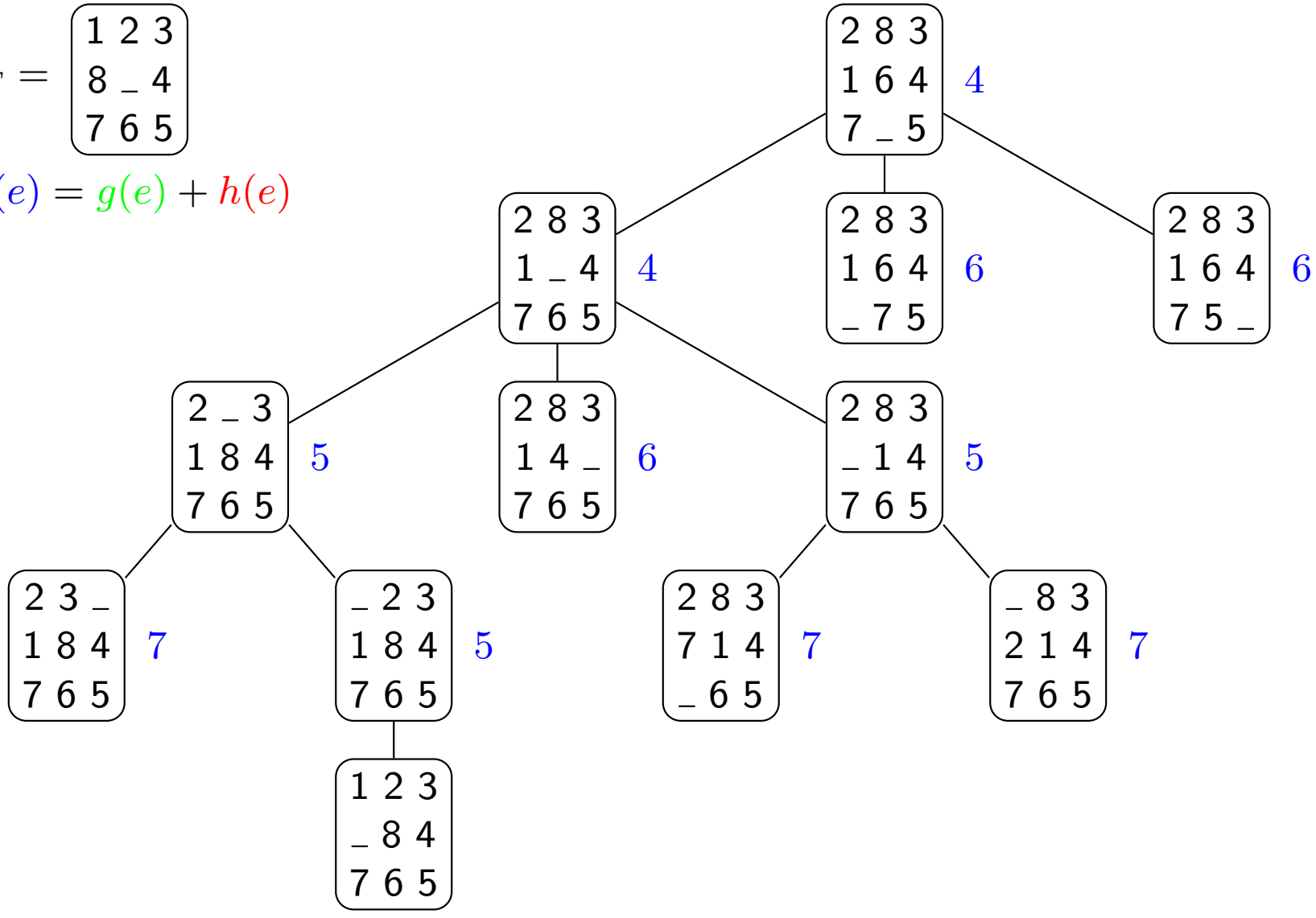


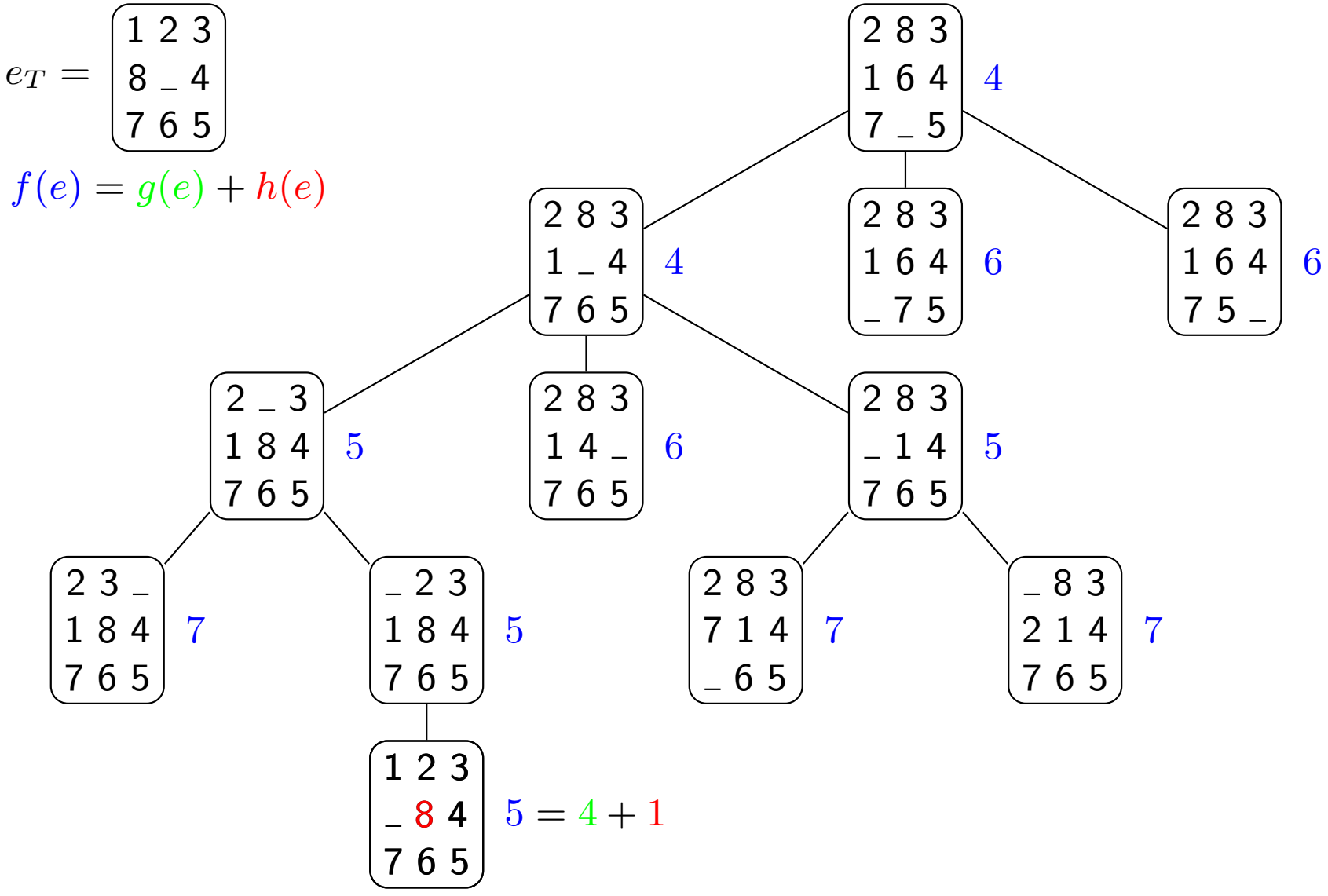

$e_T =$
1 2 3
8 _ 4
7 6 5
$f(e) = g(e) + h(e)$
2 8 3
1 6 4
7 _ 5
4
2 8 3
1 _ 4
7 6 5
4
2 8 3
1 6 4
_ 7 5
6
2 8 3
1 6 4
7 5 _
6
2 _ 3
1 8 4
7 6 5
5
2 8 3
1 4 _
7 6 5
6
2 8 3
_ 1 4
7 6 5
5
2 3 _
1 8 4
7 6 5
$7 = 3 + 4$
_ 2 3
1 8 4
7 6 5
$5 = 3 + 2$
2 8 3
7 1 4
_ 6 5
7
_ 8 3
2 1 4
7 6 5
7

$e_T =$

| | | |
|---|---|---|
| 1 | 2 | 3 |
| 8 | _ | 4 |
| 7 | 6 | 5 |

$f(e) = g(e) + h(e)$

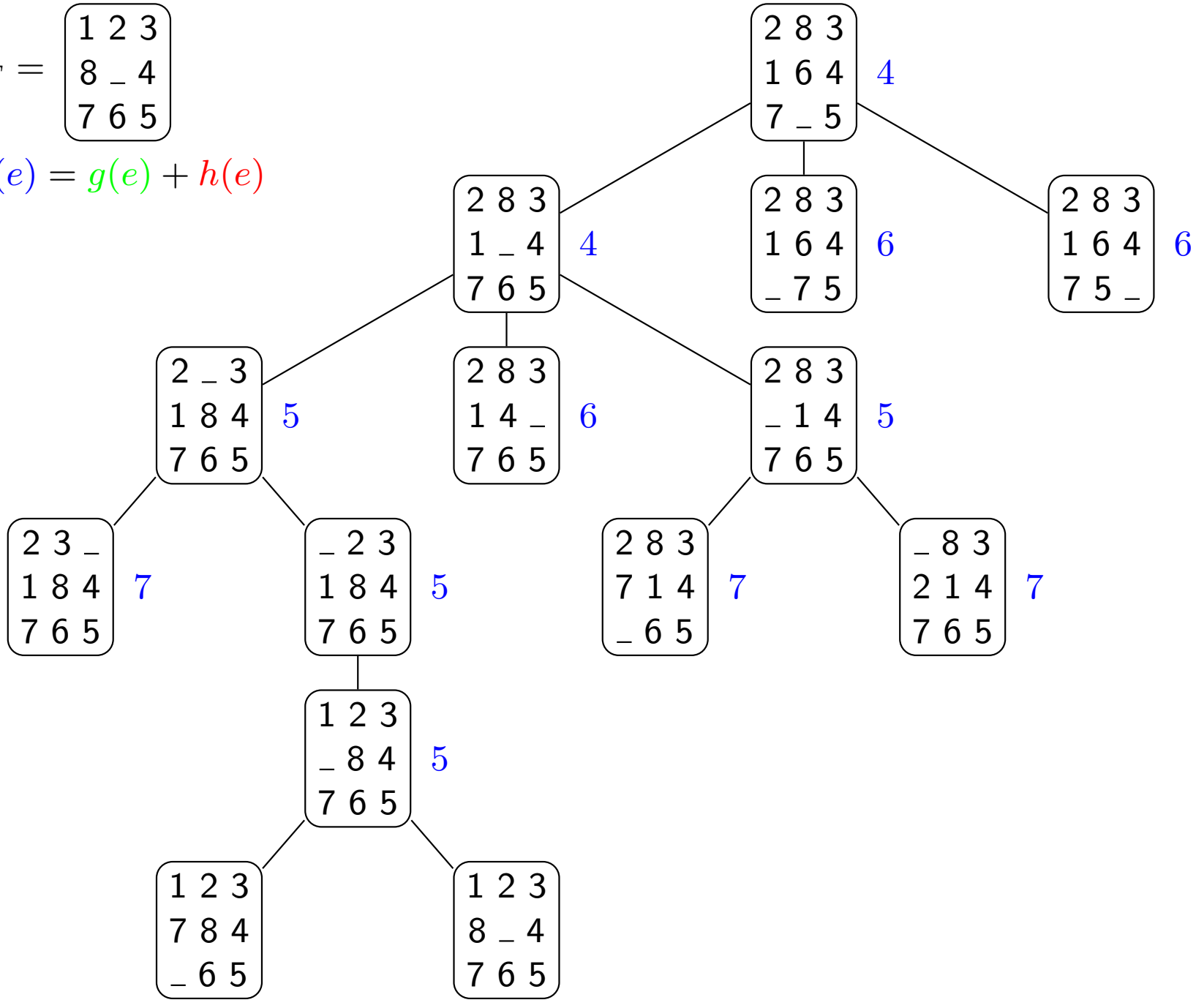


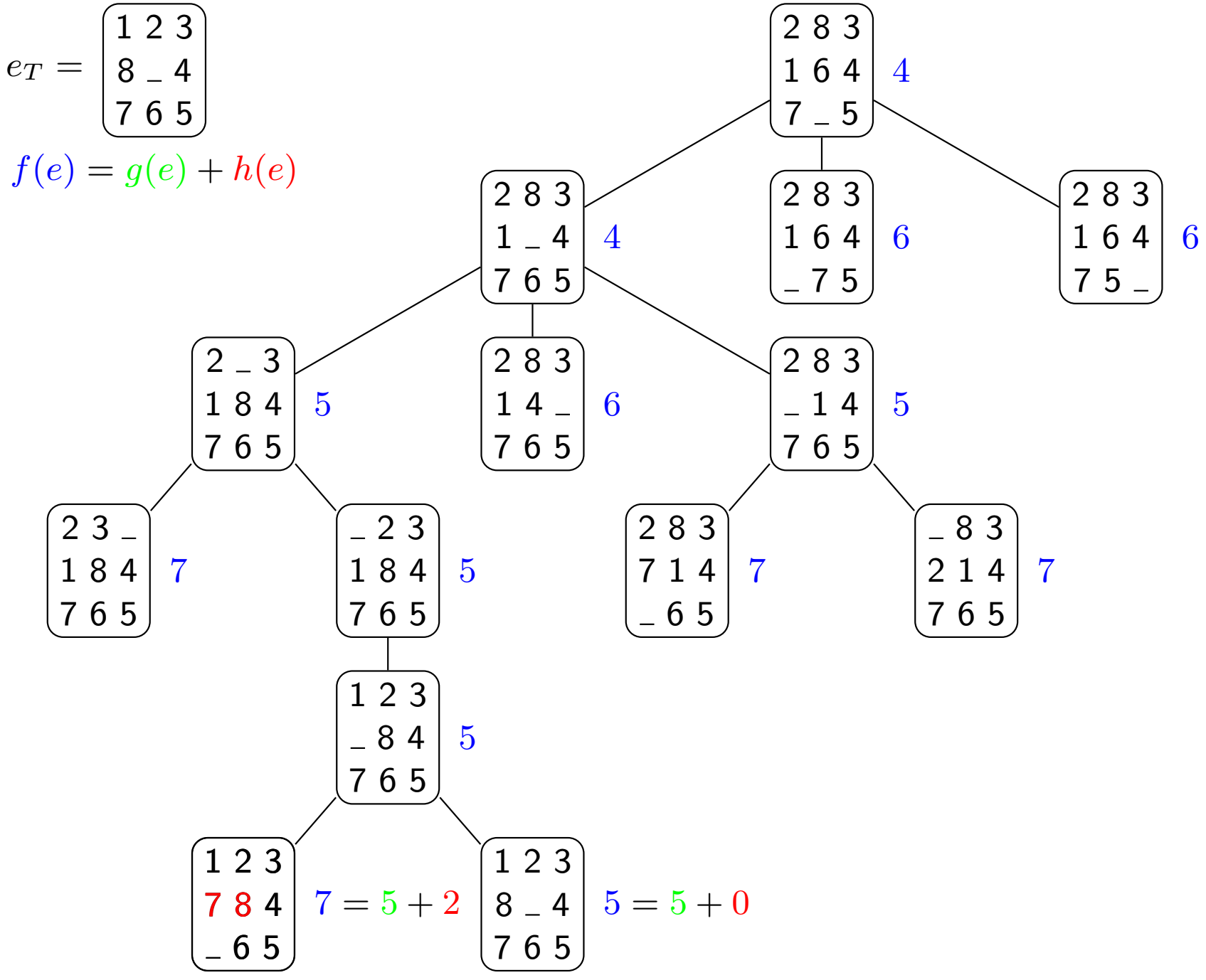

$e_T =$
1 2 3
8 _ 4
7 6 5
$f(e) = g(e) + h(e)$
2 8 3
1 6 4
7 _ 5
4
2 8 3
1 _ 4
7 6 5
4
2 8 3
1 6 4
_ 7 5
6
2 8 3
1 6 4
7 5 _
6
2 _ 3
1 8 4
7 6 5
5
2 8 3
1 4 _
7 6 5
6
2 8 3
_ 1 4
7 6 5
5
2 3 _
1 8 4
7 6 5
7
_ 2 3
1 8 4
7 6 5
5
2 8 3
7 1 4
_ 6 5
7
_ 8 3
2 1 4
7 6 5
7
1 2 3
_ 8 4
7 6 5
$5 = 4 + 1$

$e_T =$

| | | |
|---|---|---|
| 1 | 2 | 3 |
| 8 | _ | 4 |
| 7 | 6 | 5 |

$f(e) = g(e) + h(e)$




$e_T =$
1 2 3
8 _ 4
7 6 5
$f(e) = g(e) + h(e)$
2 8 3
1 6 4
7 _ 5
4
2 8 3
1 _ 4
7 6 5
4
2 8 3
1 6 4
_ 7 5
6
2 8 3
1 6 4
7 5 _
6
2 _ 3
1 8 4
7 6 5
5
2 8 3
1 4 _
7 6 5
6
2 8 3
_ 1 4
7 6 5
5
2 3 _
1 8 4
7 6 5
7
_ 2 3
1 8 4
7 6 5
5
2 8 3
7 1 4
_ 6 5
7
_ 8 3
2 1 4
7 6 5
7
1 2 3
_ 8 4
7 6 5
5
1 2 3
7 8 4
_ 6 5
$7 = 5 + 2$
1 2 3
8 _ 4
7 6 5
$5 = 5 + 0$

$e_T =$

| | | |
|---|---|---|
| 1 | 2 | 3 |
| 8 | _ | 4 |
| 7 | 6 | 5 |

$f(e) = g(e) + h(e)$