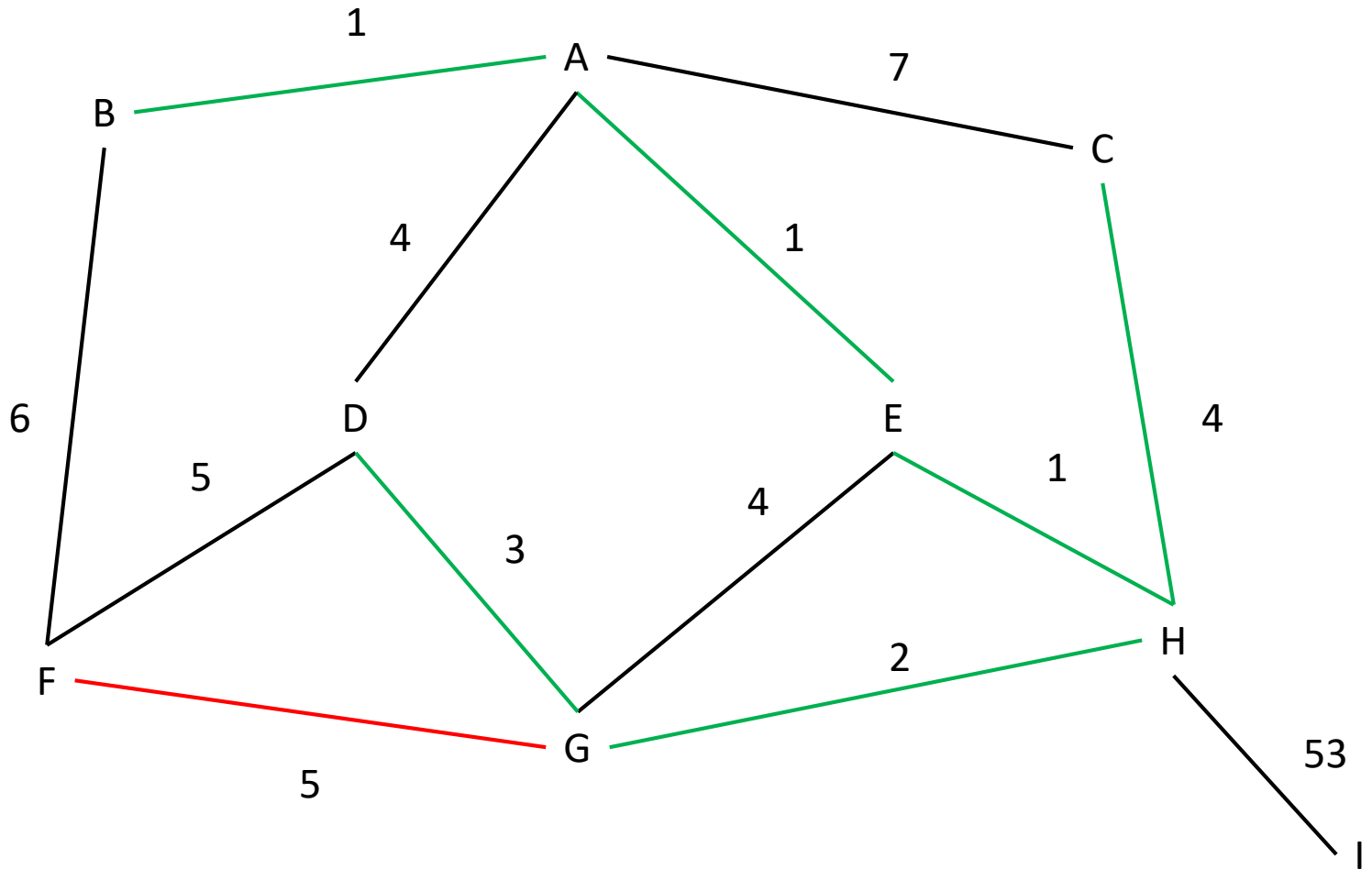
ARBRE COUVRANT DE POIDS MINIMAL PARTIE 2 : ALGORITHME DE PRIM

Algorithme de Prim (Idées)

- On choisit un sommet s de départ
- A l'étape i nous avons construit un arbre enraciné en s contenant i arêtes.
- On choisit une arête de poids minimal susceptible d'agrandir notre arbre
- On a un arbre contenant i+1 arêtes (prêt pour l'étape i+1)

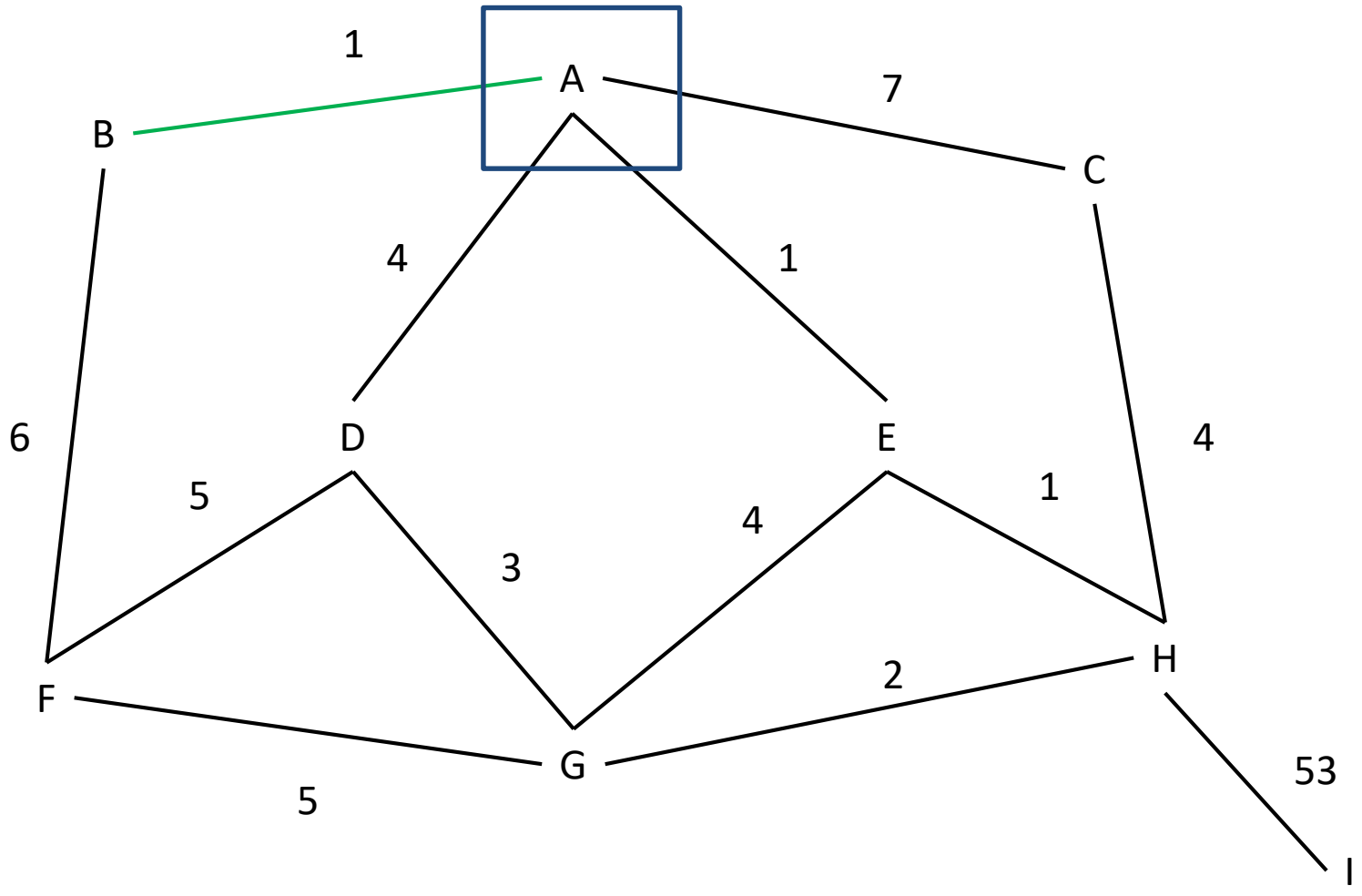
Exemple d'exécution



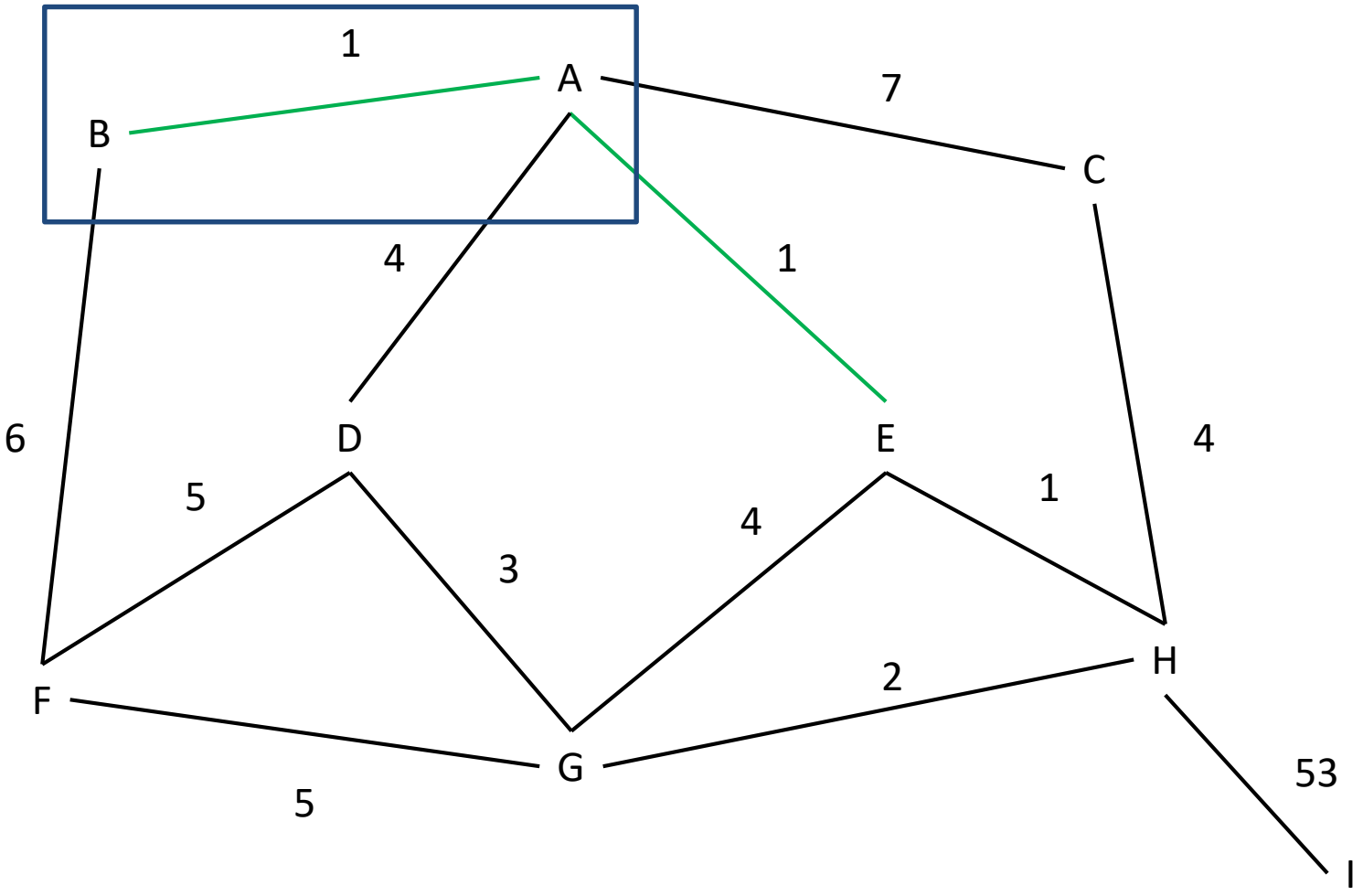
Algorithme de Prim (Idées)

- Il est fondé sur le premier principe
- On choisit un sommet s de départ
- Initialement : Set = {s}; U’ = {}
- Tant que Set <> X
  - Choisir une arête uv telle que :
    - Une et une seule des deux extrémités appartient à Set.
    - Cette arête uv est de poids minimal
  - Insérer uv dans U’; Set ← Set U {u, v}

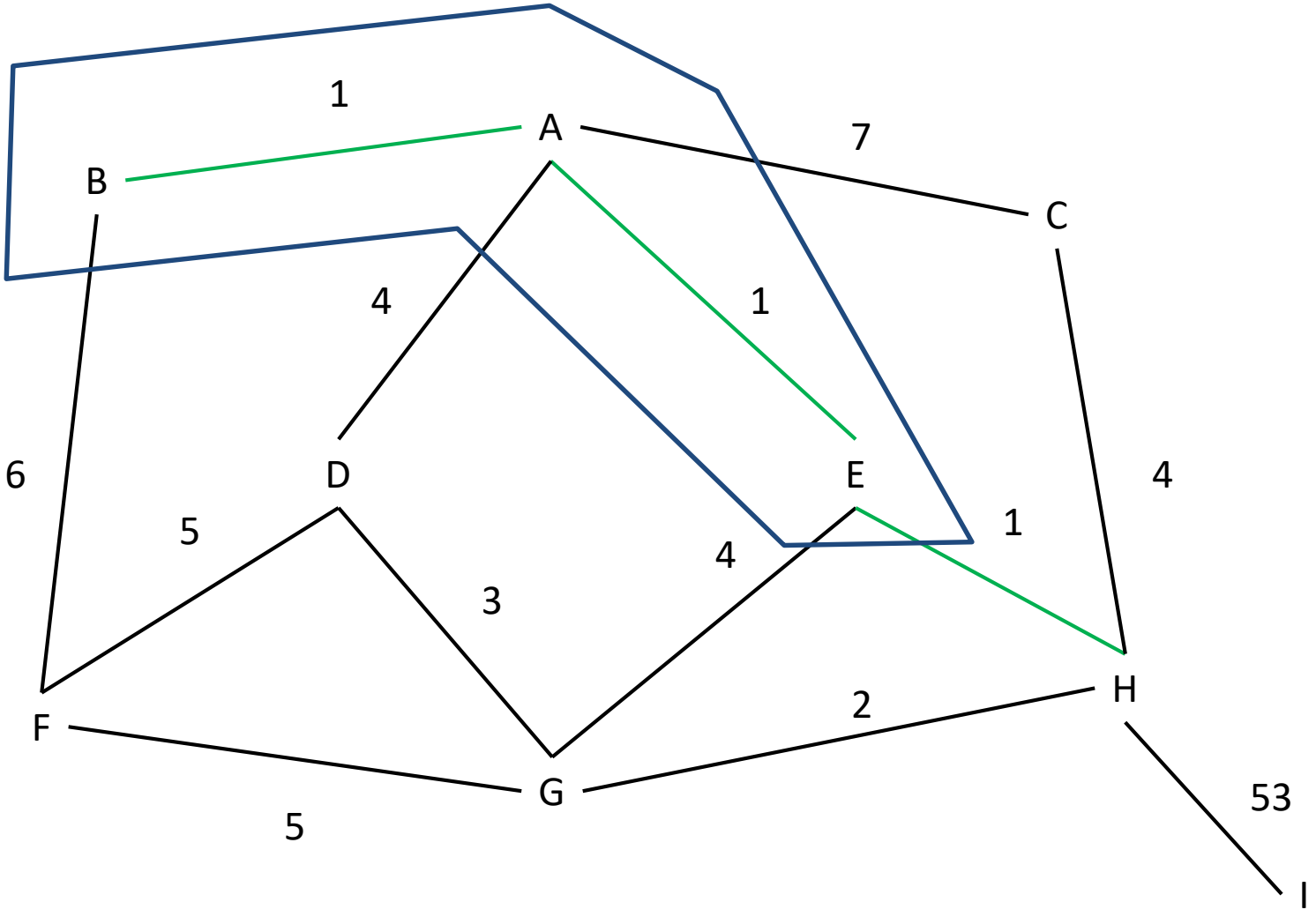
Exemple d'exécution



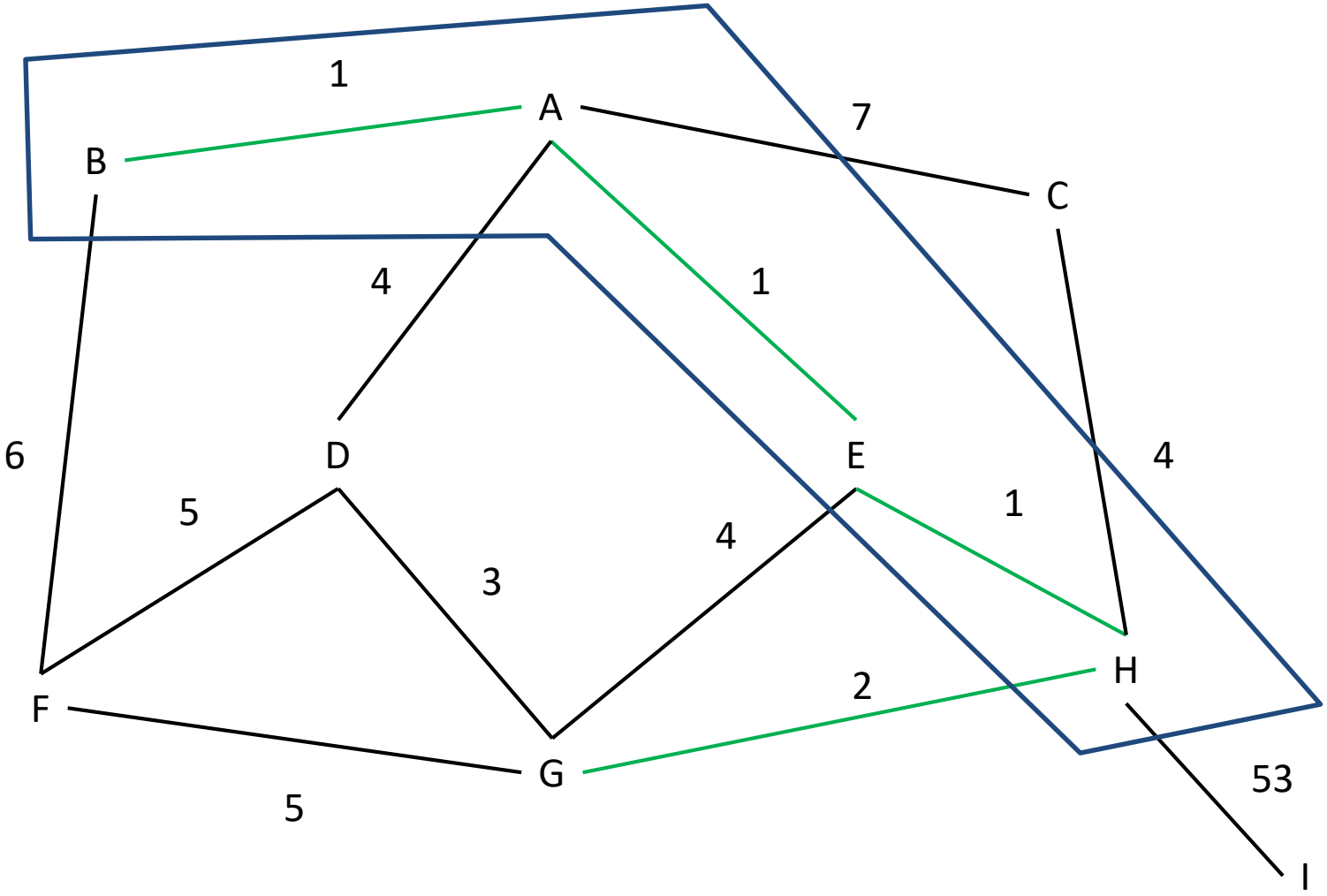
Exemple d'exécution



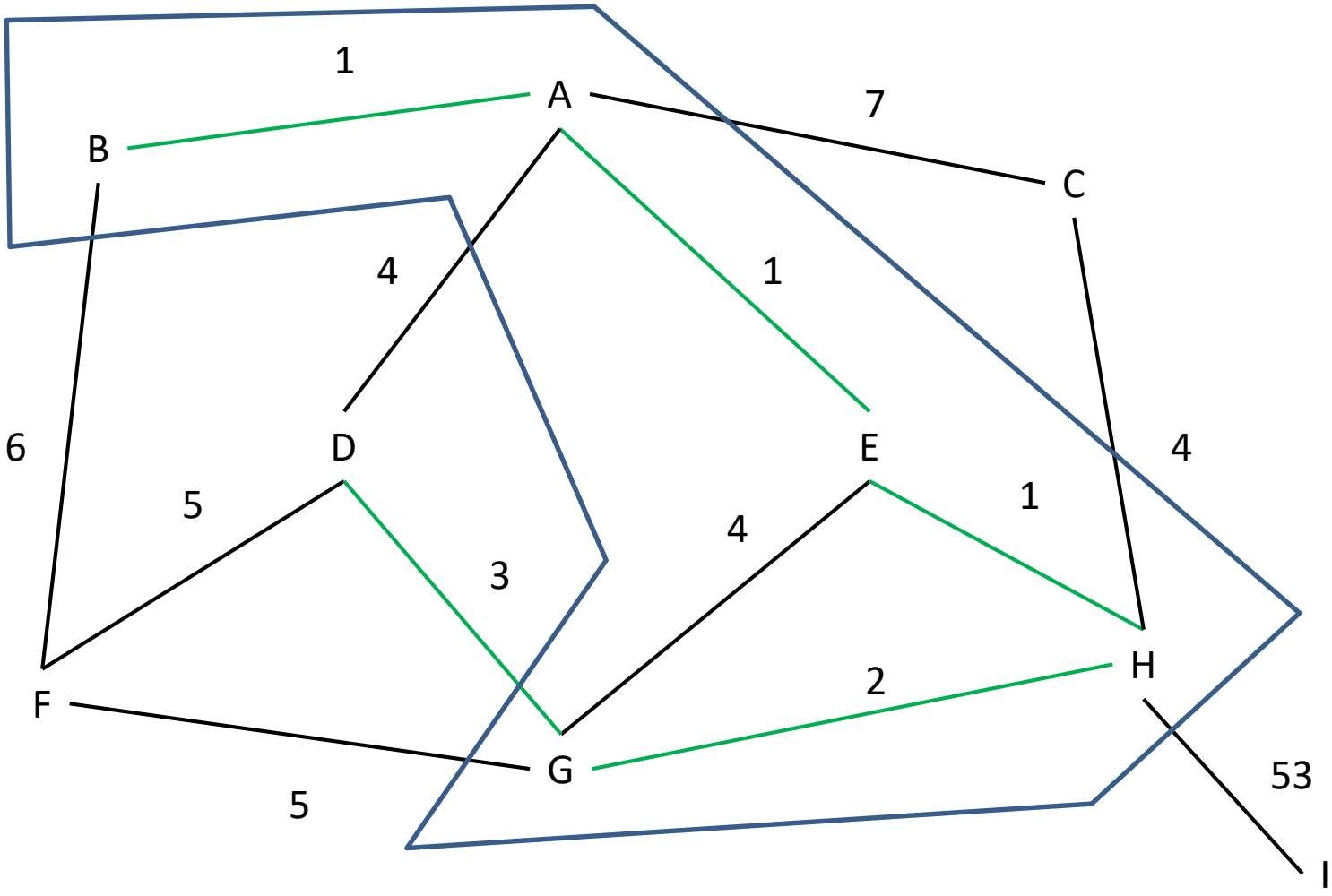
Exemple d'exécution



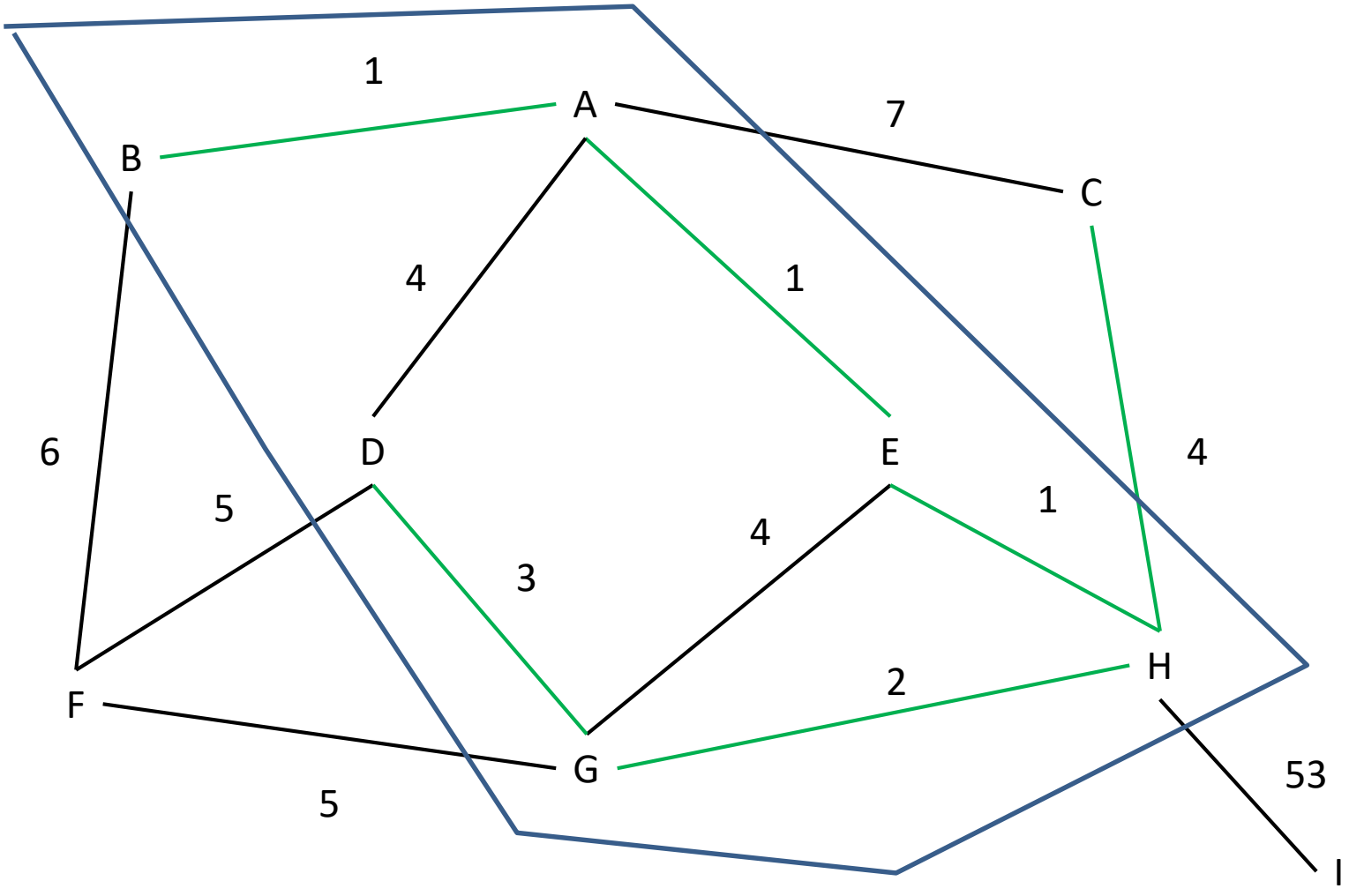
Exemple d'exécution



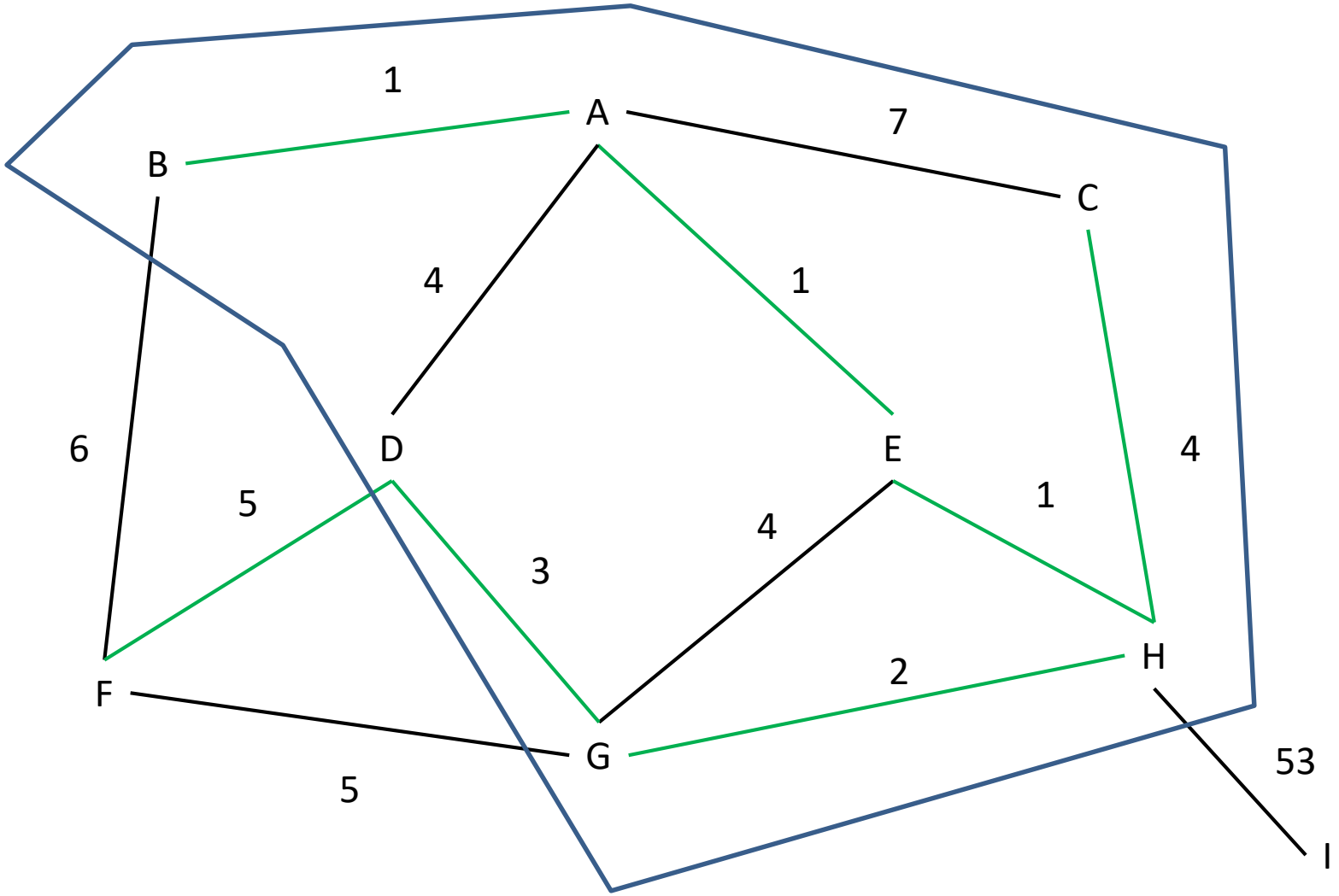
Exemple d'exécution



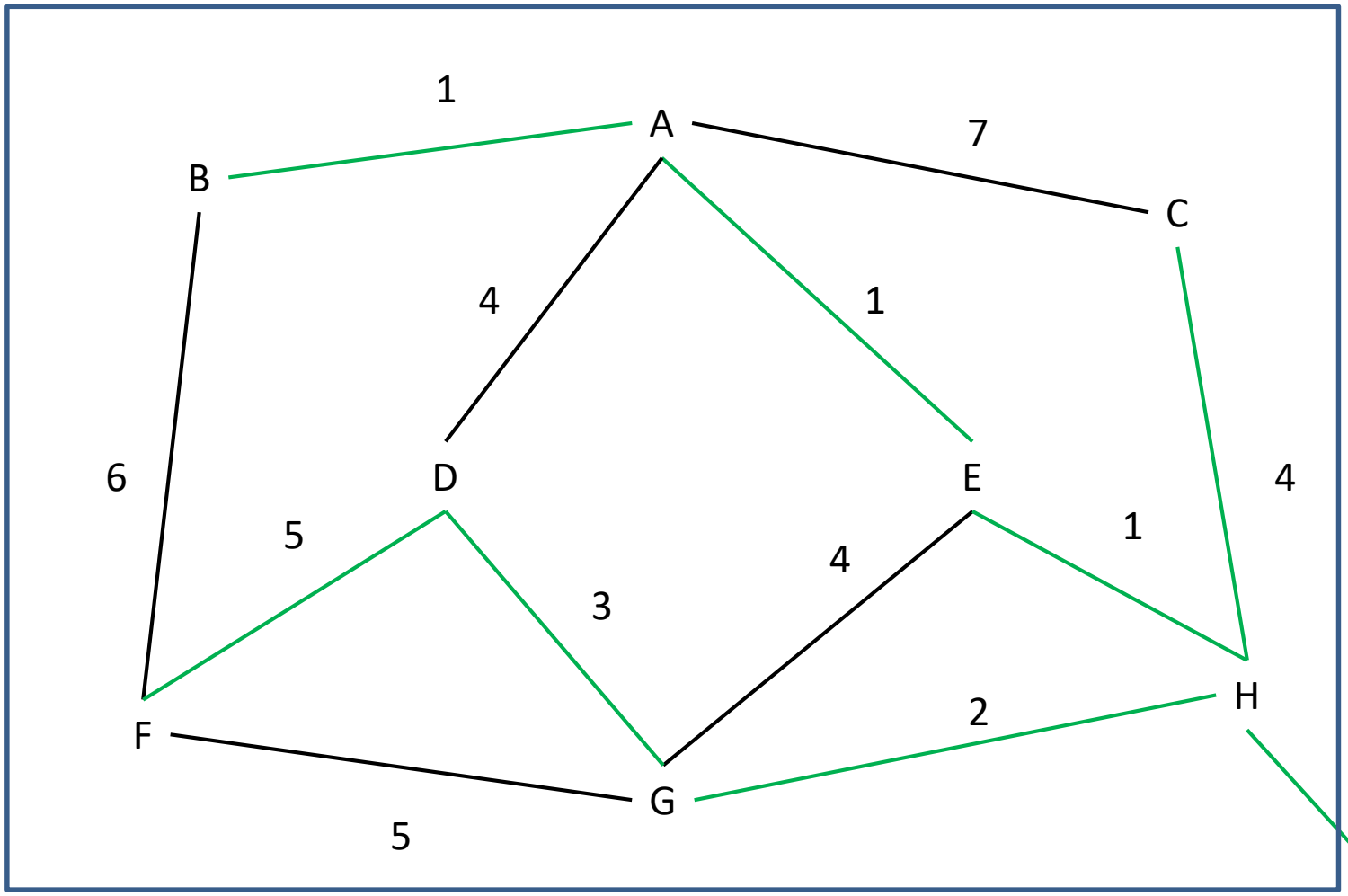
Exemple d'exécution



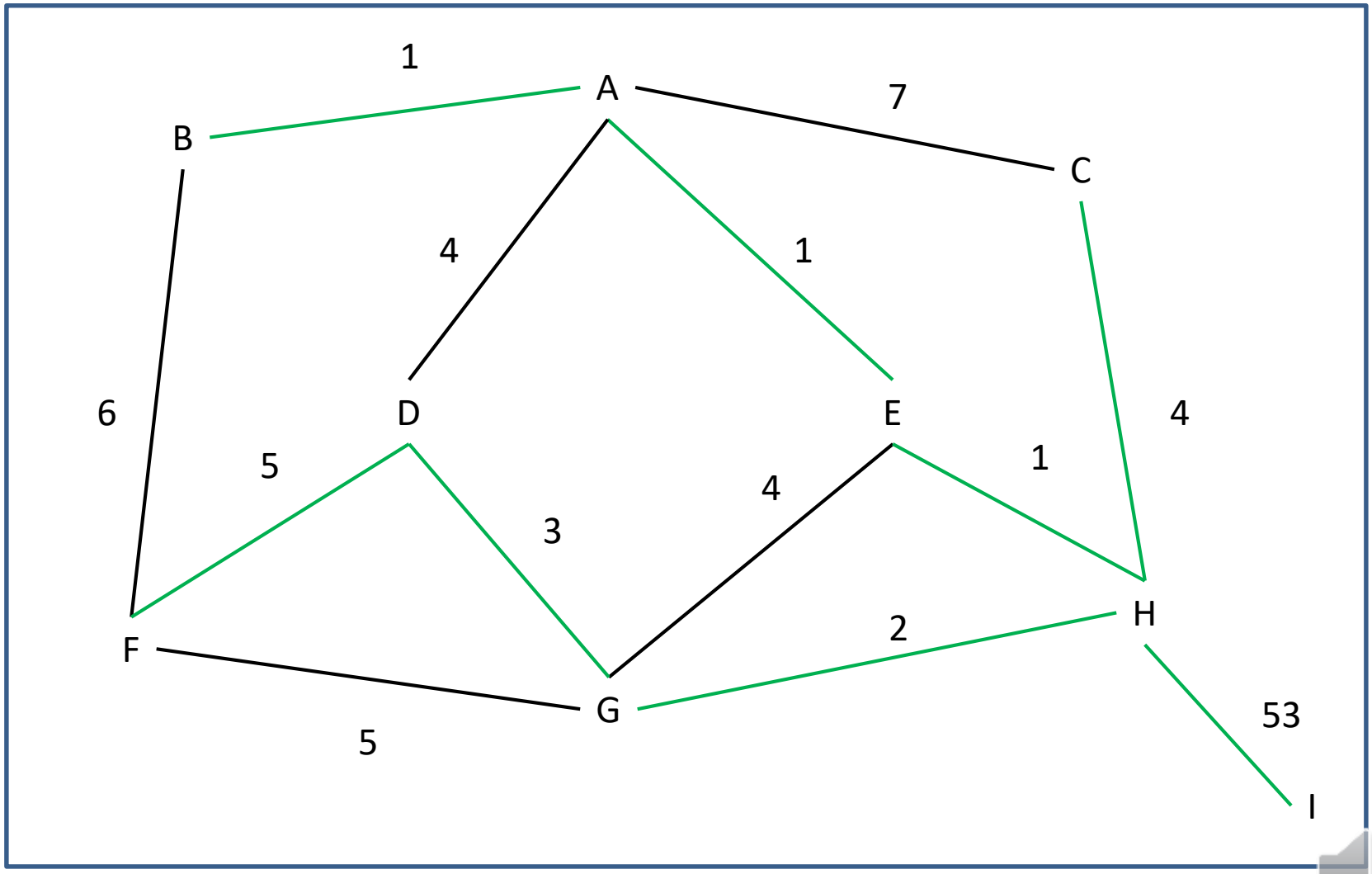
Exemple d'exécution



Exemple d'exécution



Exemple d'exécution



Algorithme de Prim (Entête)

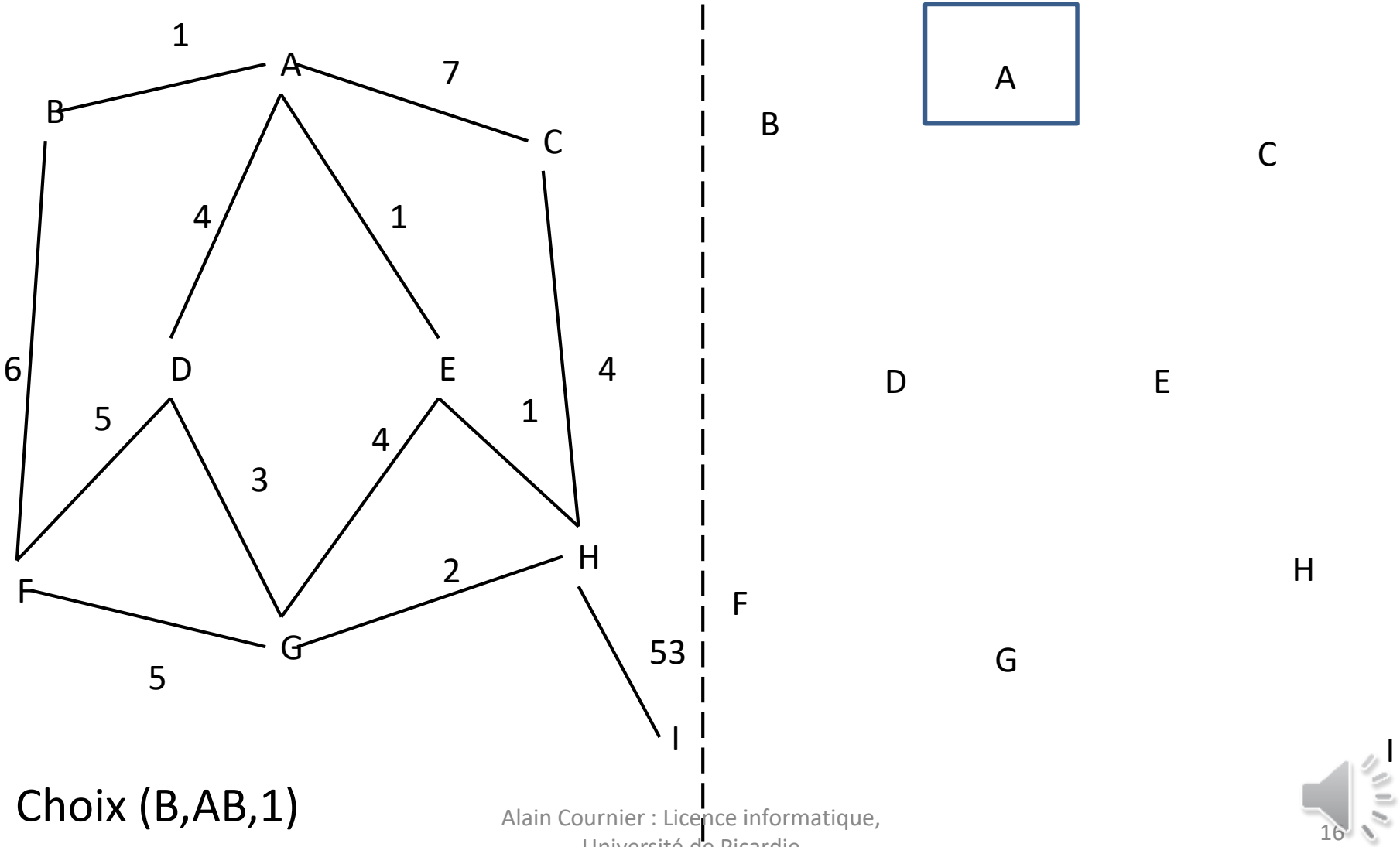
- Données :
  - G = (X, U, V) un graphe pondéré
  - x un sommet de G
- Résultat : ACPM = (S, A, V) un graphe pondéré
  - Initialement S = X; A = {};
- Variables :
  - Set : ensemble de sommets Initialement : {}
  - Inter : ensemble de triplet (s, a, p) initialement {}
    - s est un sommet
    - a est une arête dont s est une extrémité
    - p est le poids de l'arête a

Algorithme de Prim (Code)

- Début
  - Set ← {x}
  - Pour chaque voisin de x (y,V(x,y)) faire Insérer (y, xy, V(xy)) dans Inter
  - FinPour
  - Tant Que Inter <> Vide faire
    - Choisir (y, zy, p) tel que p minimal dans Inter; Retirer (y, zy, p) de Inter;
    - Si non (y ϵ Set) alors
      - Set ← Set U {y}, A ← A U {zy}
      - Pour chaque voisin de y (t,p') faire
        - » Si non (t ϵ Set) alors Insérer (t, yt, p') dans Inter
      - Finpour
    - FinSi
  - FinTant Que
- Fin

Inter = {(B,AB,1),(D,AD,4),(E,AE,1),(C,AC,7)}

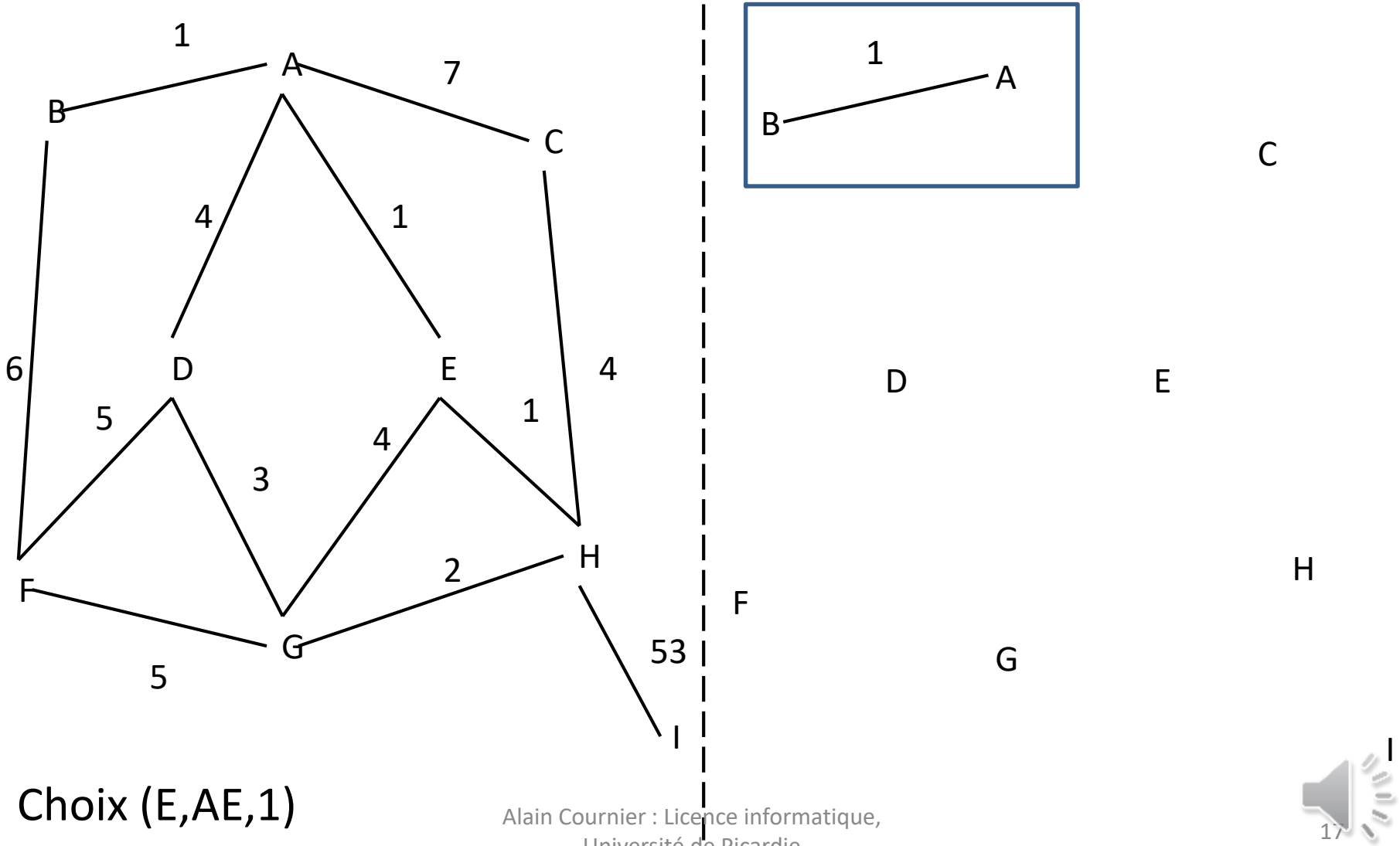
Set = {A} représenté par le rectangle bleu



Choix (B,AB,1)

Inter = {(D,AD,4),(E,AE,1),(C,AC,7),(F,BF,6)}

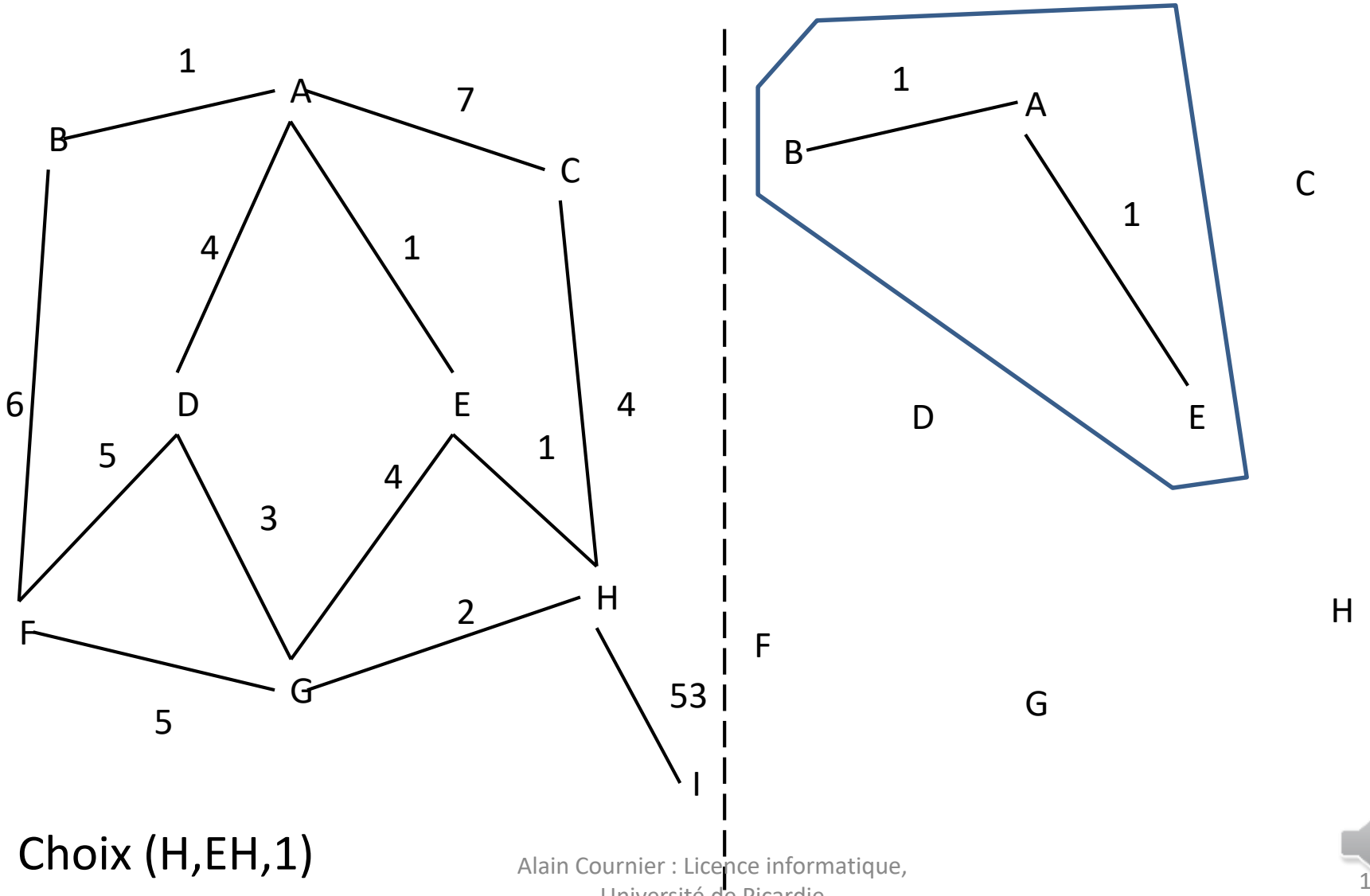
Set = {A,B}



Choix (E,AE,1)

Inter = {(D,AD,4),(C,AC,7),(F,BF,6),(H,EH,1),(G,EG,4)}

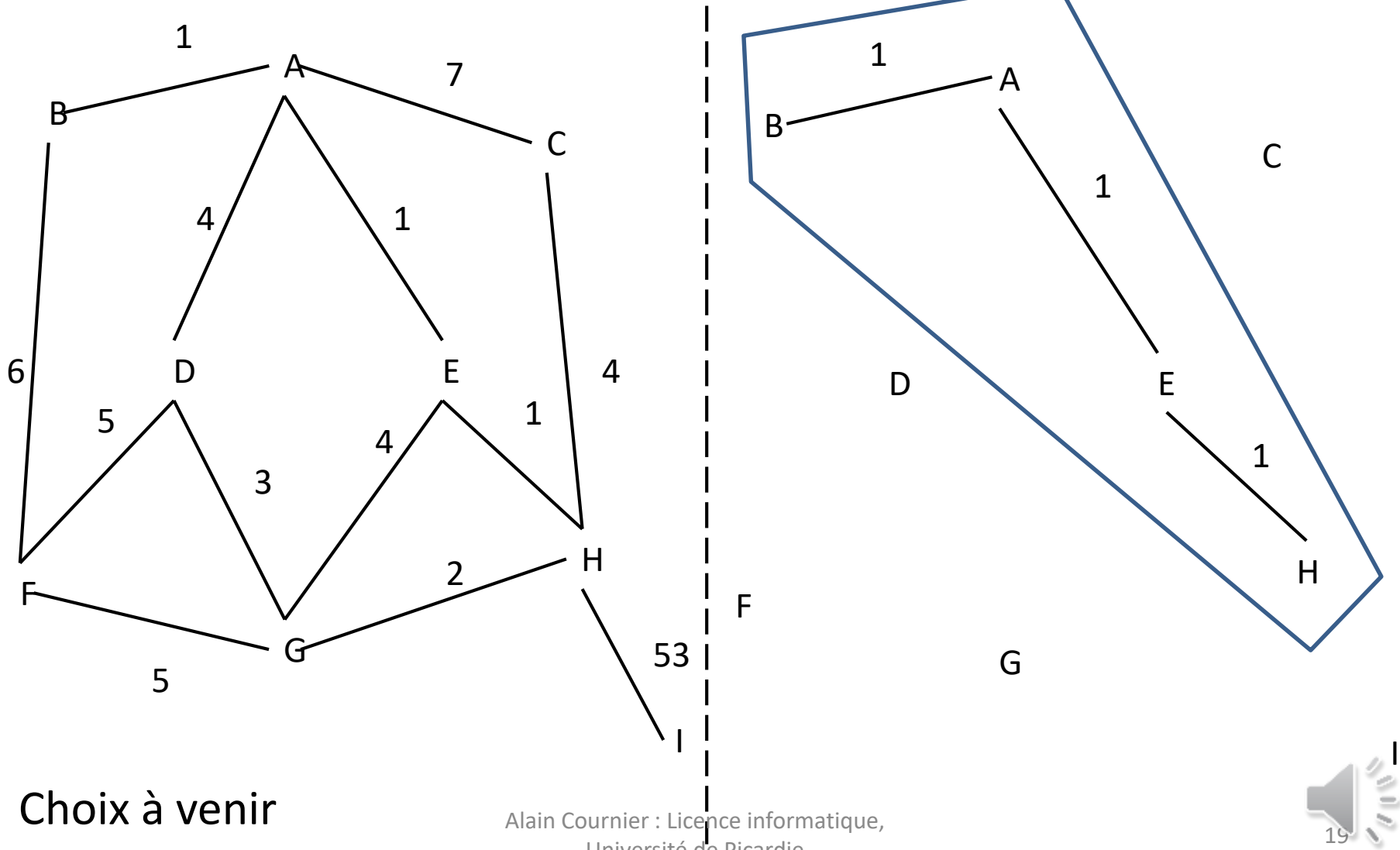
Set = {A,B,E}



Choix (H,EH,1)

Inter = {(D,AD,4),(C,AC,7),(F,BF,6),(G,EG, 4),(I,HI,53),(G,HG,2),(C,HC,4)}

Set = {A,B,E,H}



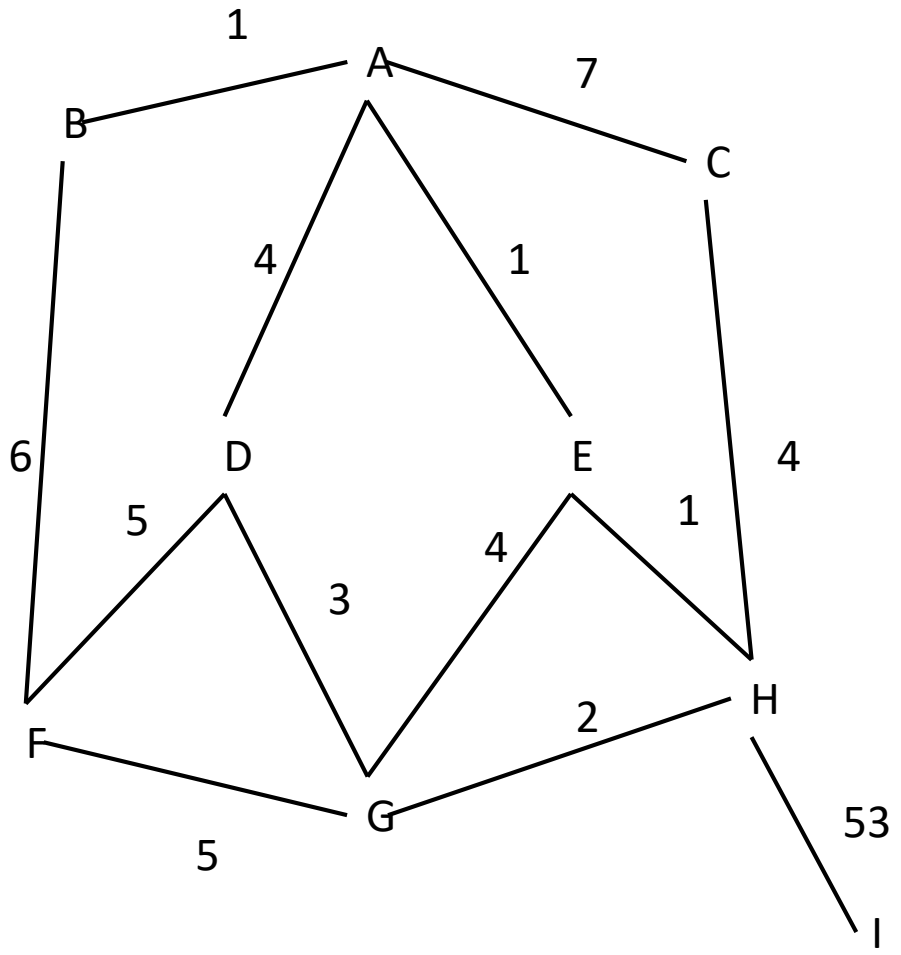
Choix à venir

Question

- Faut-il conserver les deux triplets (C,AC,7) et (C,HC,4) dans Inter ?
- Même question pour les deux triplets (G,EG,4) et (G,HG,2) ?

- Réponse non dans les deux cas on peut supprimer le triplet de poids maximal

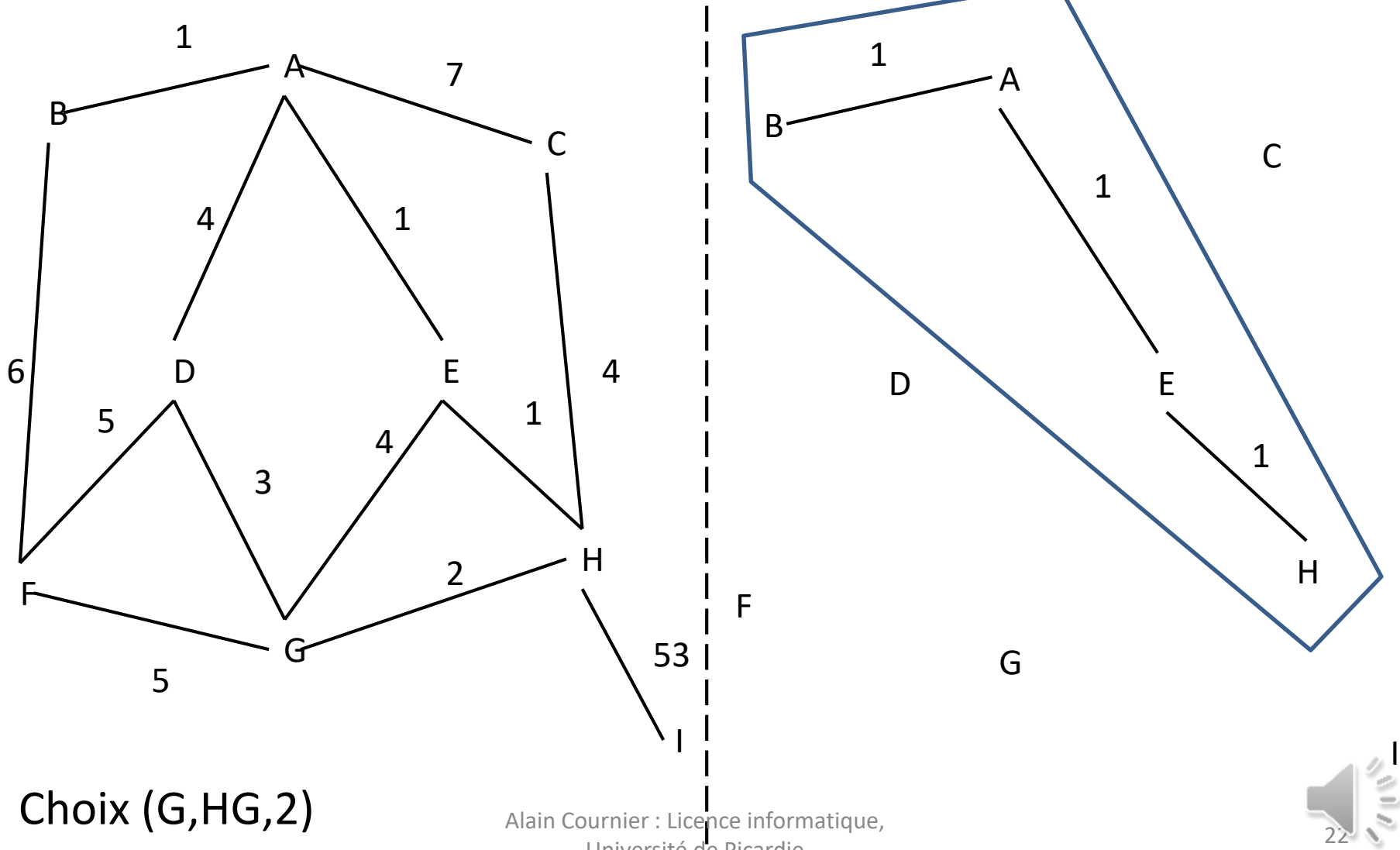
Explication



- AEHCA forme un cycle où AC est l'arête de poids maximal. Donc il est possible de l'enlever en vertu du principe 2
- EHGE forme un cycle où EG est l'arête de poids maximal.

Inter = {(D,AD,4),(F,BF,6),(I,HI,53),(G,HG,2),(C,HC,4)}

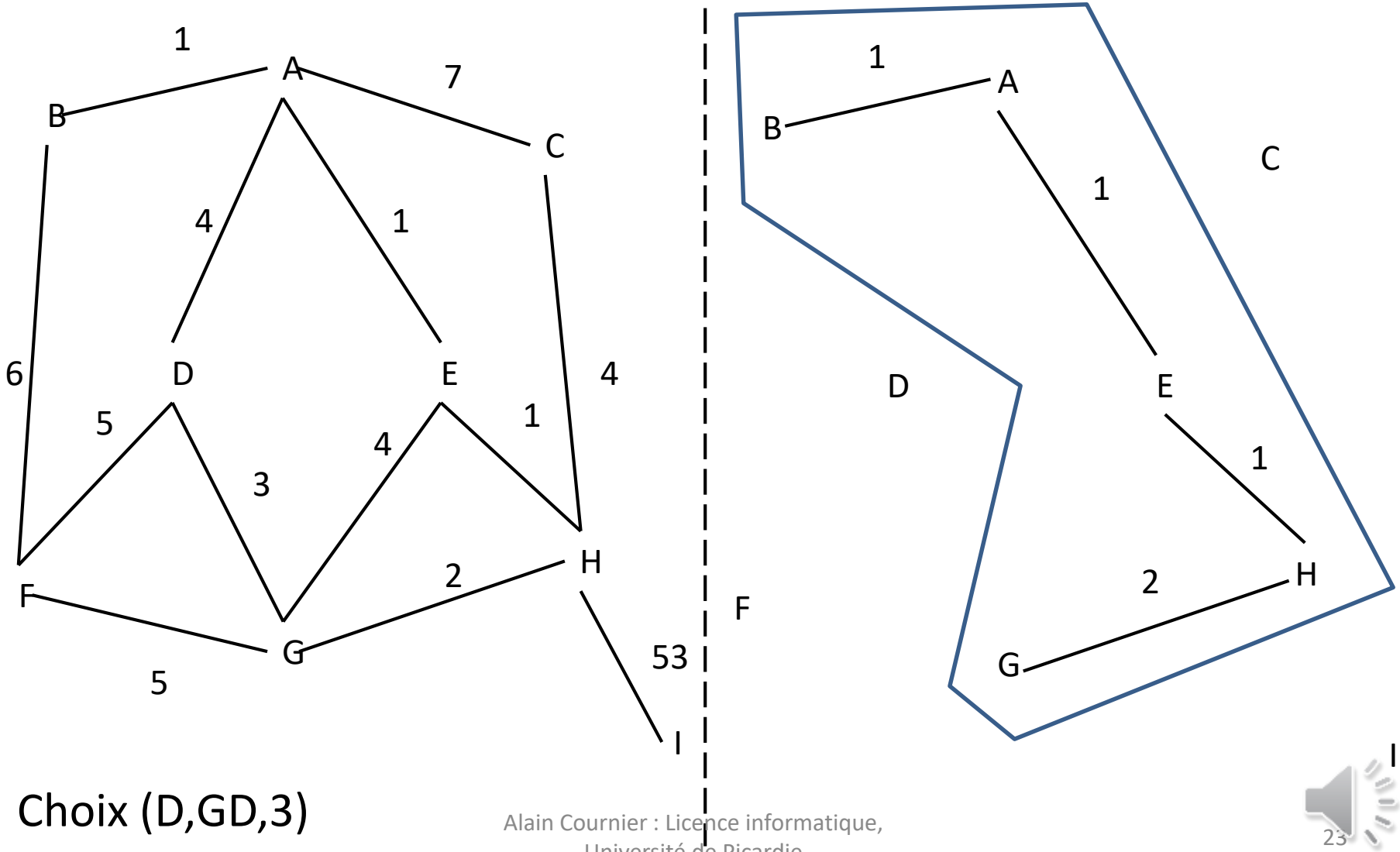
Set = {A,B,E,H}



Choix (G,HG,2)

Inter = {(D,~~AD~~,4),(F,~~BF~~,6),(I,HI,53),(C,HC,4),(D,GD,3),(F,GF,5)}

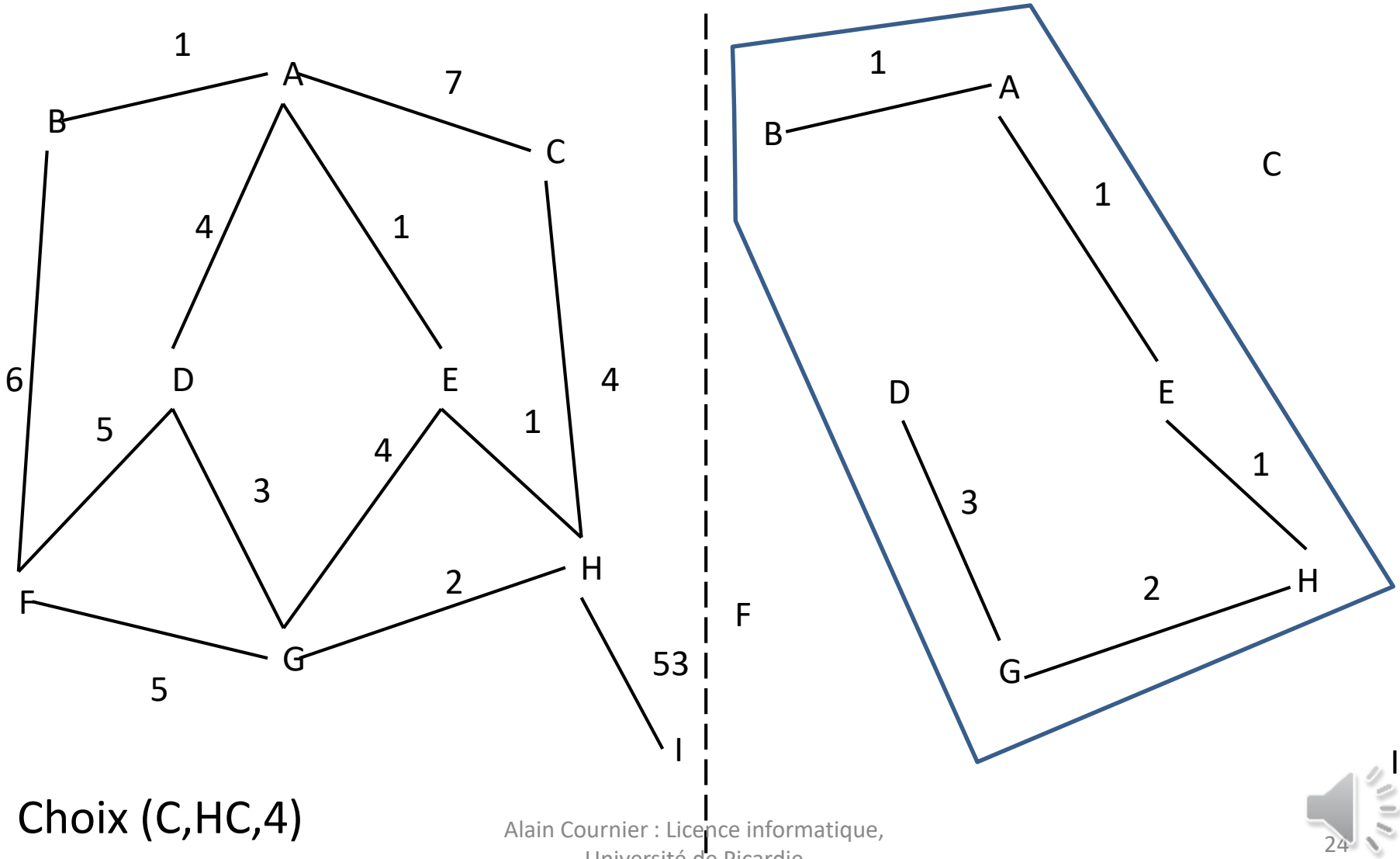
Set = {A,B,E,H,G}



Choix (D,GD,3)

Inter = {(I,HI,53),(C,HC,4),(F,GF,5),(F,DF,5)}

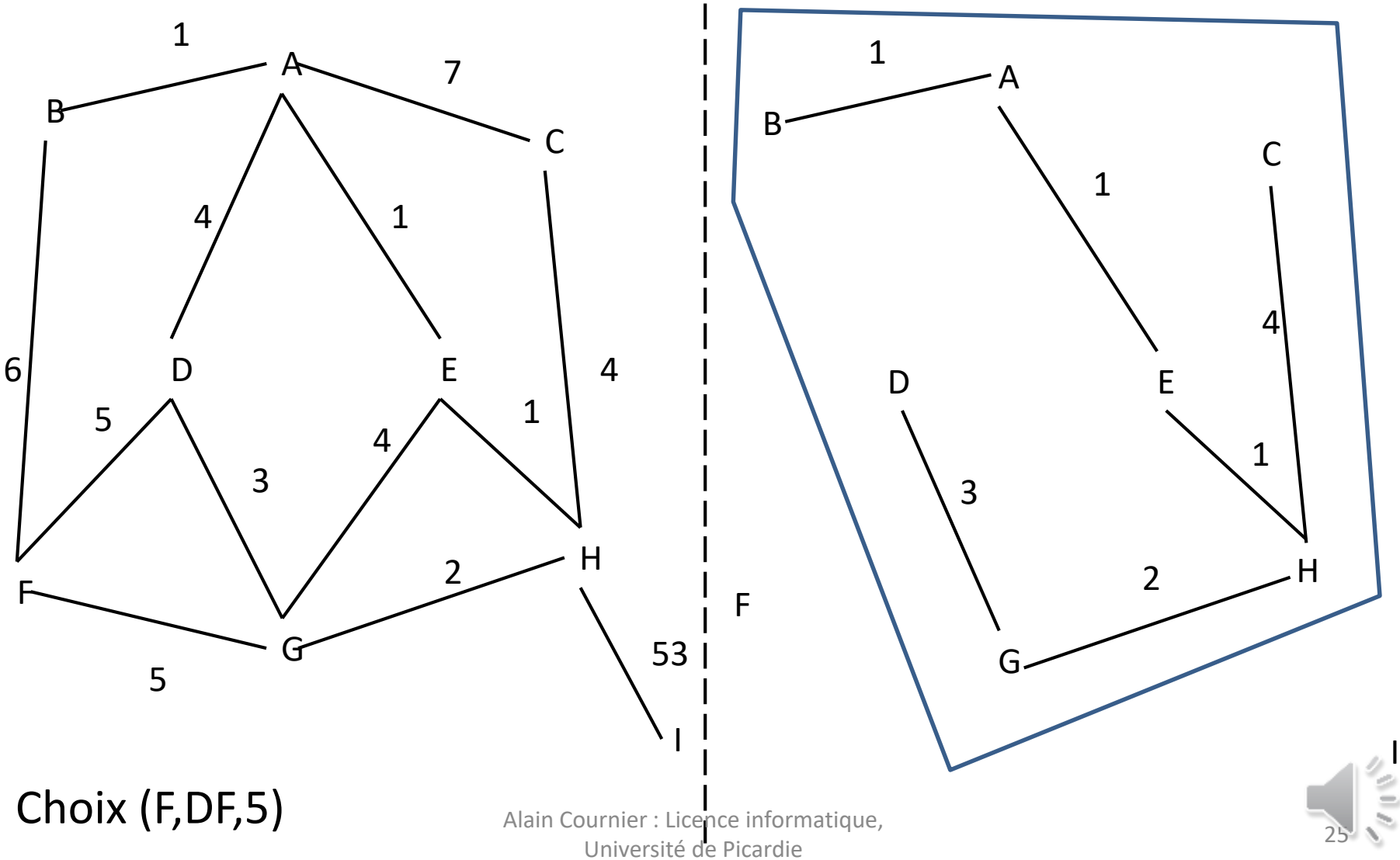
Set = {A,B,E,H,G,D}



Choix (C,HC,4)

Inter = {(I,HI,53),(F,DF,5)}

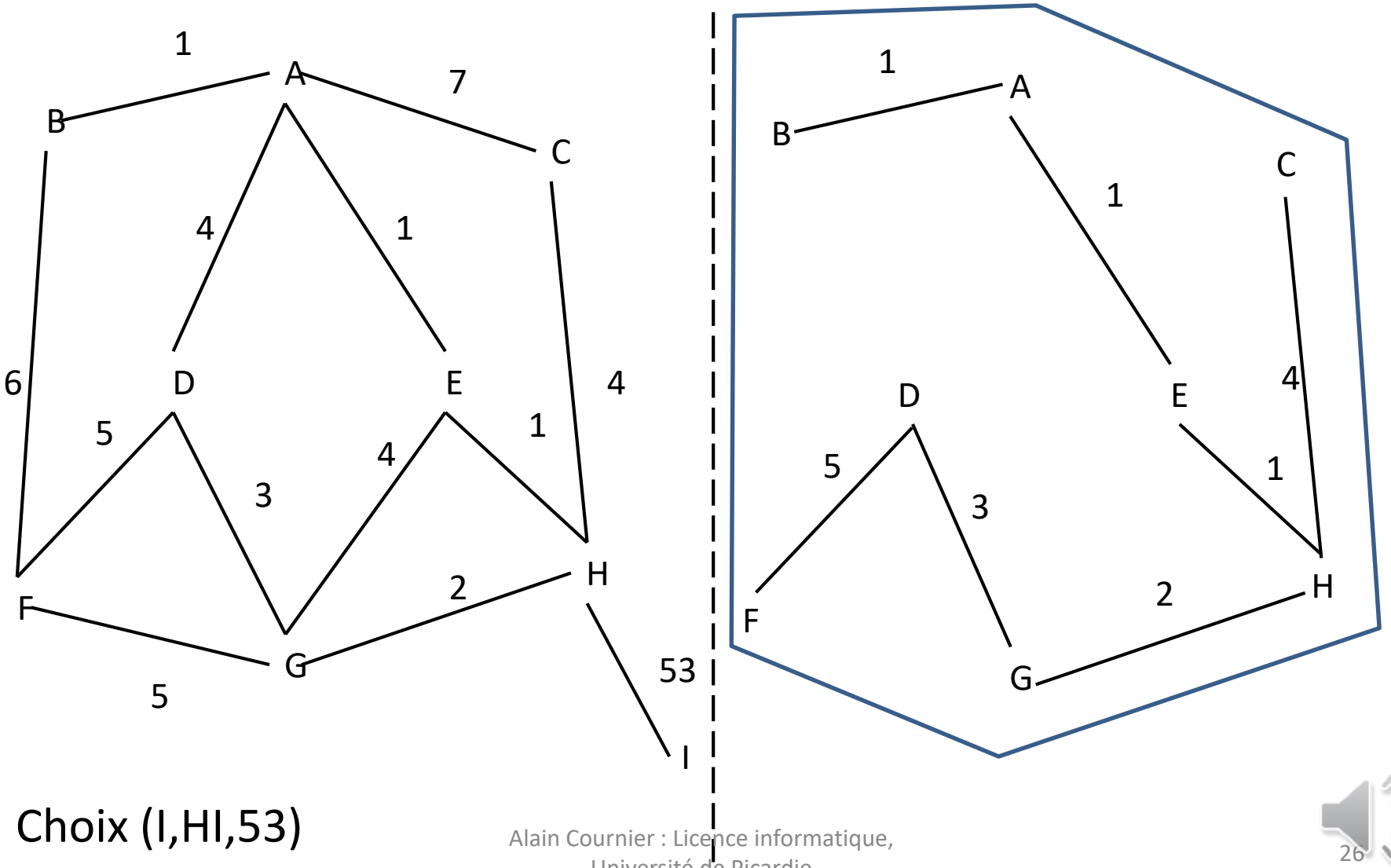
Set = {A,B,E,H,G,D,C}



Choix (F,DF,5)

Inter = {(I,HI,53)}

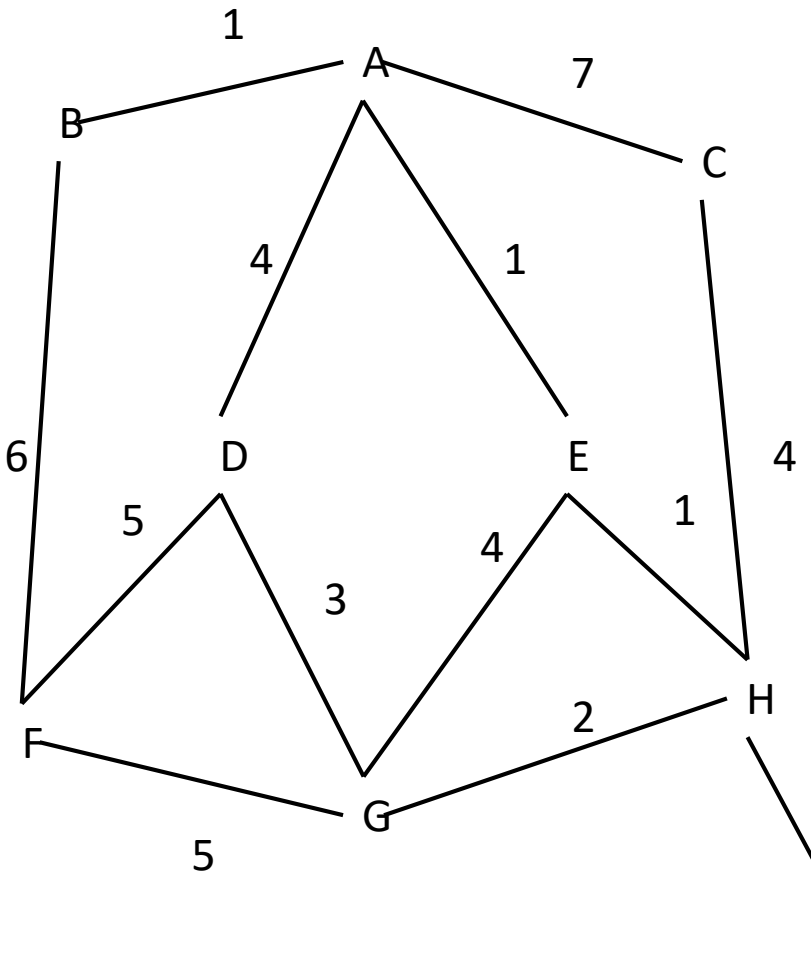
Set = {A,B,E,H,G,D,C,F}

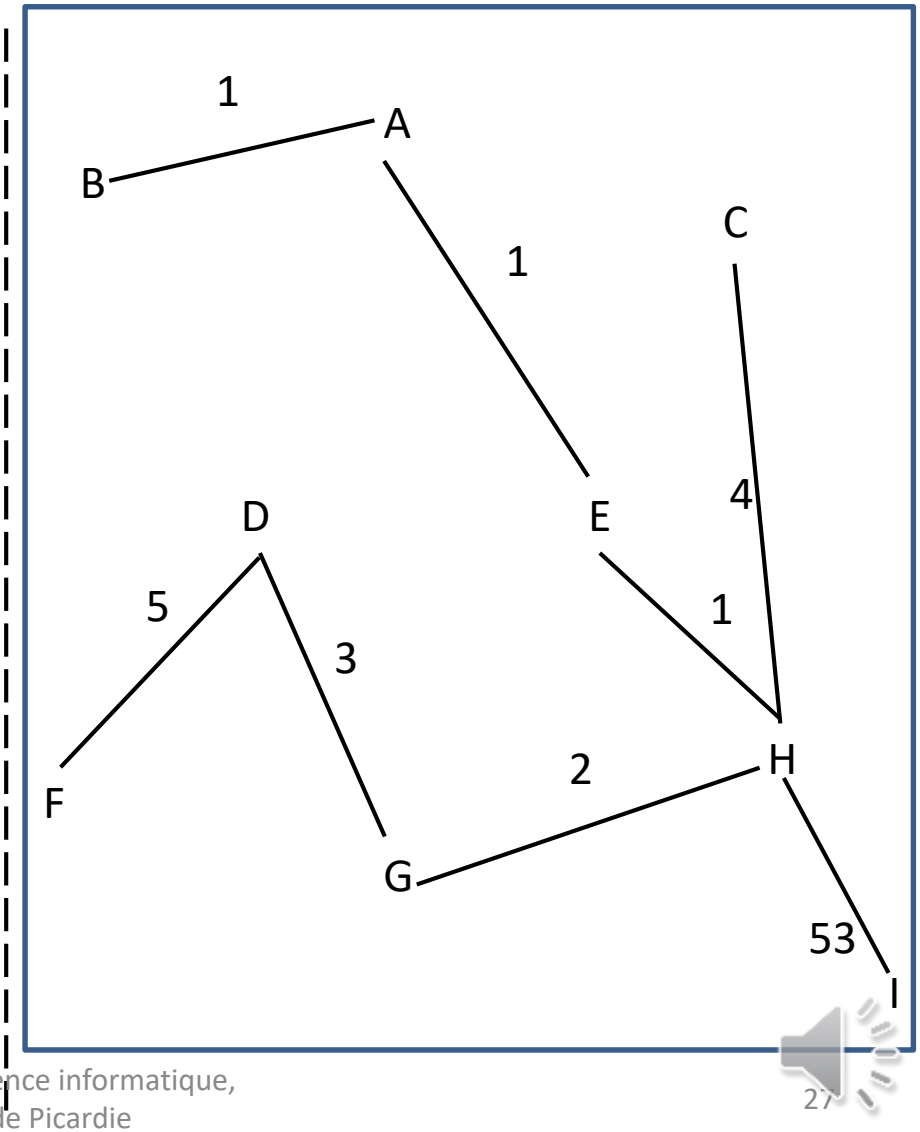


Choix (I,HI,53)

Inter = {}

Set = {A,B,E,H,G,D,C,F,I}

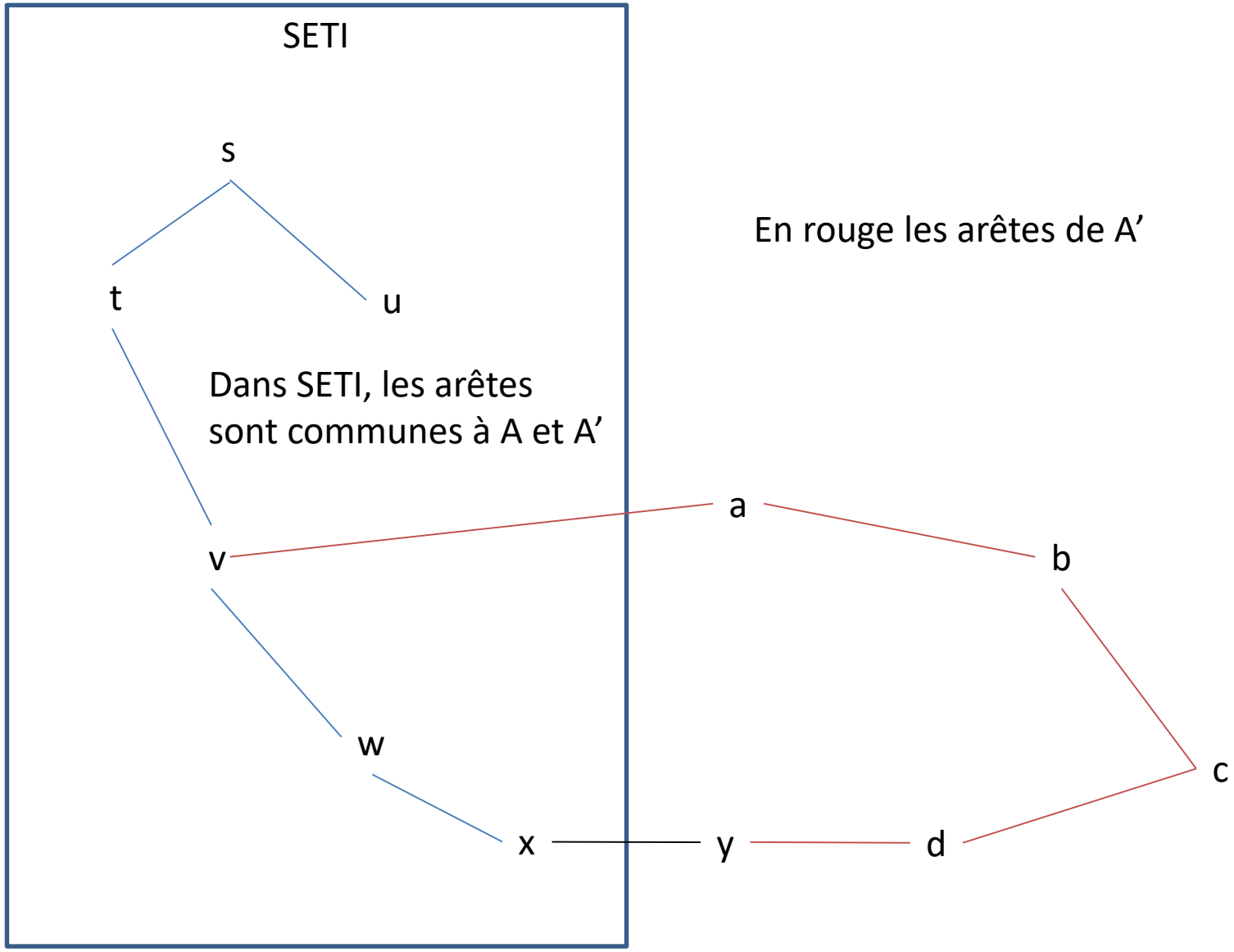




Fin de l'algorithme

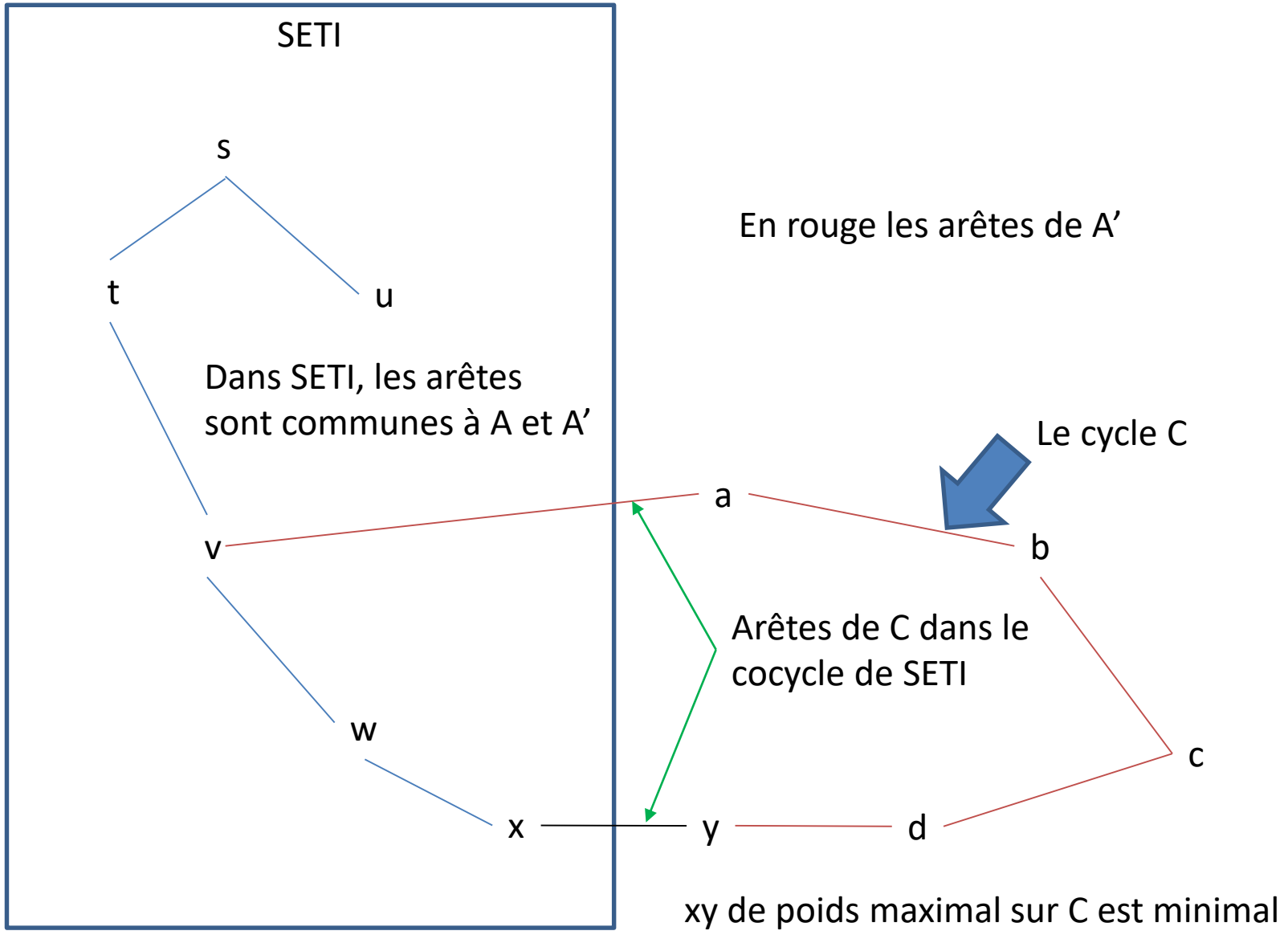
Preuve de l'algorithme

- Soit A l'arbre construit par Prim et A' un arbre couvrant de poids minimal
- Plaçons nous sur l'étape i où pour la première fois l'algorithme de Prim choisit une arête xy qui n'est pas dans A'
- Soit Seti la valeur de notre ensemble Set à l'étape i


SETI
s
t
u
Dans SETI, les arêtes
sont communes à A et A'
En rouge les arêtes de A'
a
v
b
w
c
x
y
d

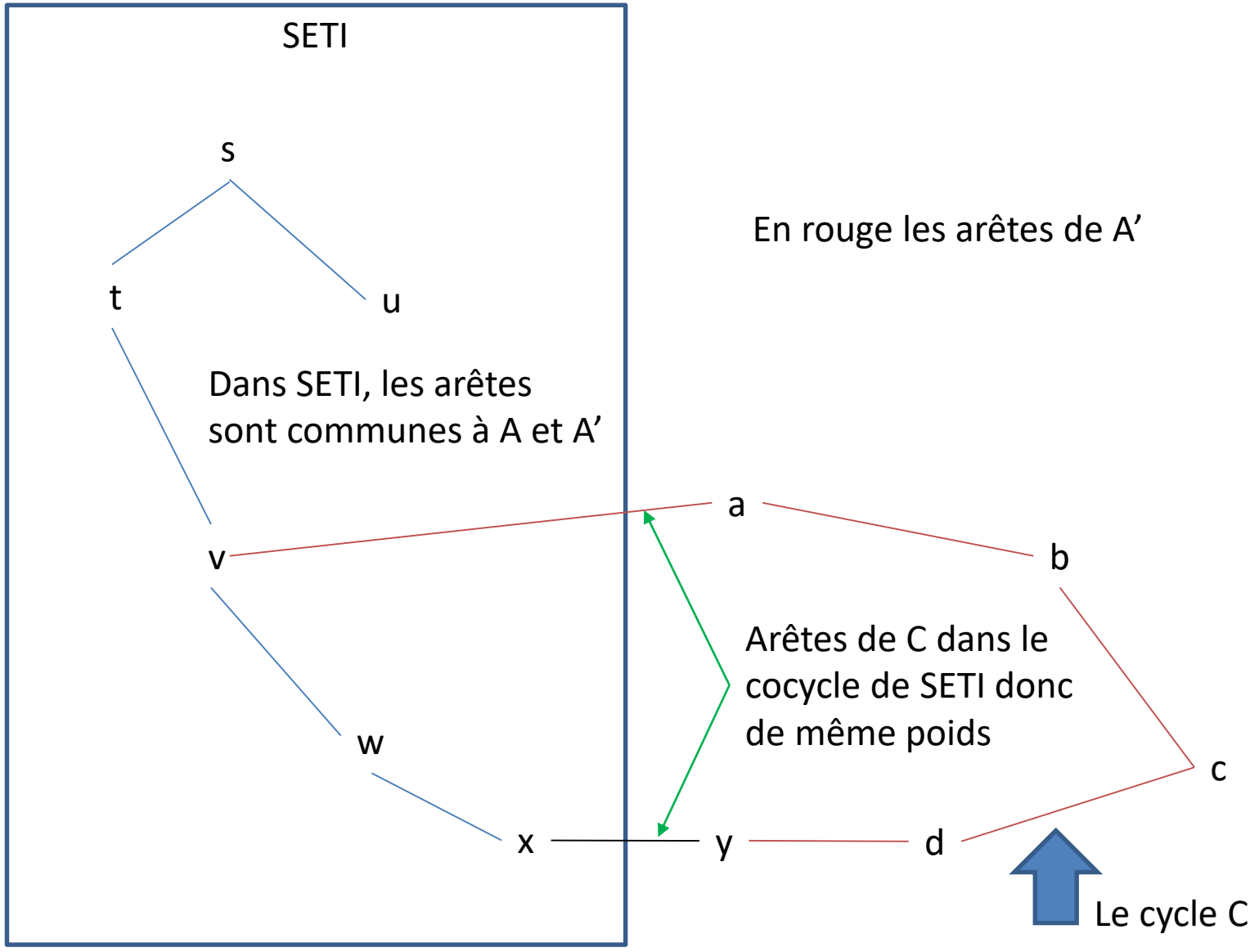
Preuve de l'algorithme

- Remarque 1 : Ajouter l'arête xy dans l'arbre A', crée un cycle C dont au moins deux arêtes sont dans le cocycle de SETI.
- Remarque 2 : Puisque A' est un arbre de poids minimal, l'arête xy est de poids maximal sur le cycle.
- Remarque 3 : L'arête xy est de poids minimal dans le cocycle de SETI.


SETI
s
t
u
Dans SETI, les arêtes
sont communes à A et A'
En rouge les arêtes de A'
Le cycle C
a
v
b
Arêtes de C dans le
cocycle de SETI
w
c
x
y
d
xy de poids maximal sur C est minimal
sur le cocycle de SETI

Preuve de l’algorithme

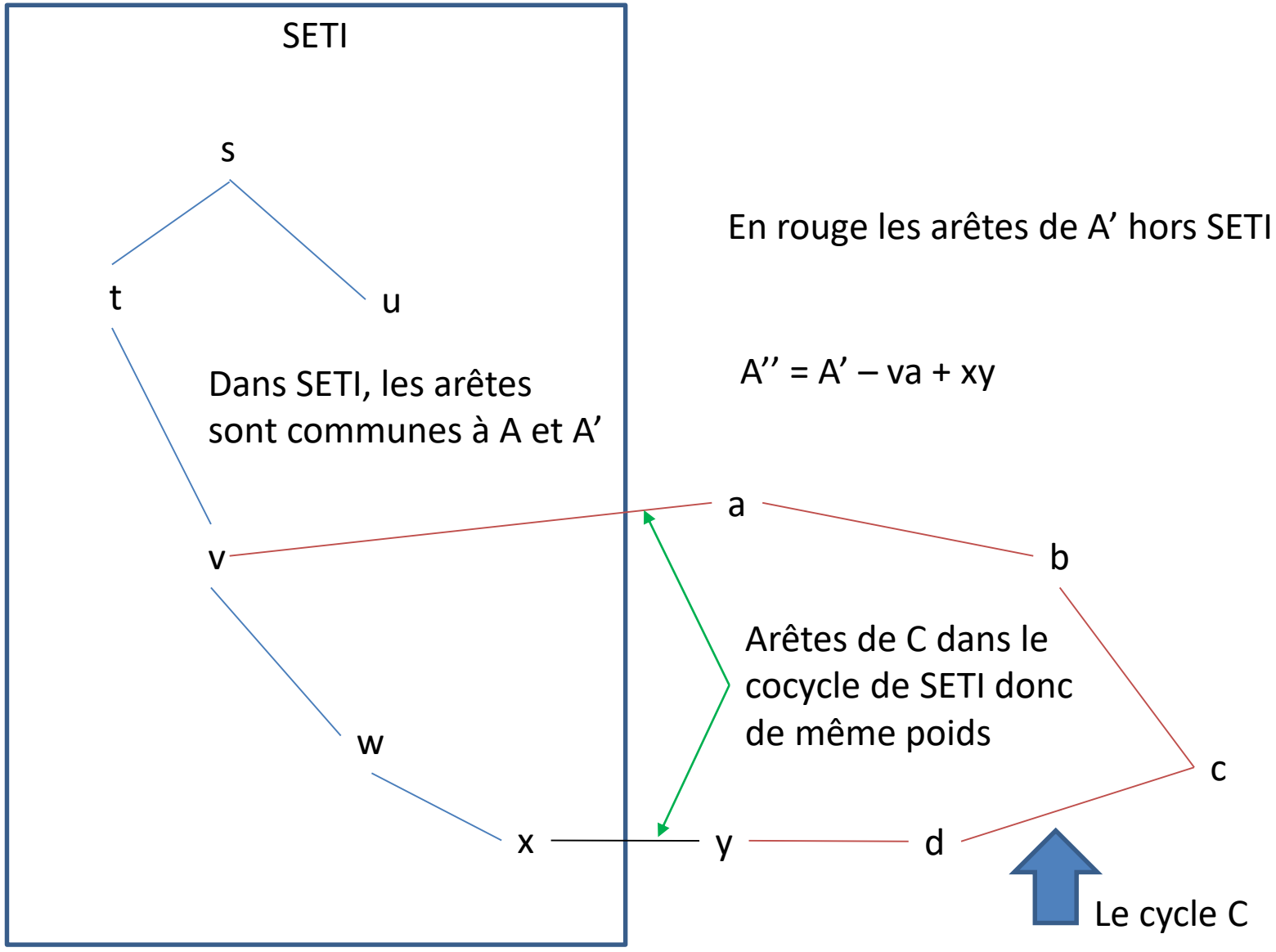
- Remarque 4 : Les arêtes du cycle C dans le cocycle de SETI ont toutes le même poids (celui de l’arête xy).
- Remarque 5 : Soit A’’ l’arbre construit à partir A’ en retirant une arête de l’intersection de A’ et de C dans le cocycle de SETI puis en ajoutant l’arête xy. A’ et A’’ sont de même poids donc A’’ est un ACPM


SETI
s
t
u
Dans SETI, les arêtes
sont communes à A et A'
En rouge les arêtes de A'
a
v
b
Arêtes de C dans le
cocycle de SETI donc
de même poids
w
c
x
y
d
Le cycle C

Preuve de l'algorithme

- Remarque 6 : L'arbre construit par notre algorithme coïncide avec l'ACPM A'' jusqu'à l'étape i+1.

- Conséquence : Après |X|-1 étapes notre algorithme de Prim a calculé un ACPM.


SETI
s
t
u
Dans SETI, les arêtes
sont communes à A et A'
En rouge les arêtes de A' hors SETI
A'' = A' – va + xy
a
v
b
Arêtes de C dans le
cocycle de SETI donc
de même poids
w
c
x
y
d
Le cycle C

Complexité

- On fait m insertions dans Inter
- On ne souhaite pas qu'un même sommet soit présent dans deux triplets différents
- On choisit au moins n-1 fois la plus petite valeur
- En utilisant une structure de données inspirée des tas binomiaux on obtient une complexité de O(m log n)
  - n nombre de sommets m nombre d'arêtes de G