Introduction aux Blockchains

16 septembre 2019

Plan du cours

7 semaines + projet

1. Aperçu général
2. Cryptographie
3. Algorithmes de consensus
4. Privacy
5. Systèmes distribués en milieu adversarial
6. Protocoles économiques
7. Soutenance de projet

Domaines dans la blockchain

- Cryptographie
- Réseau
- Bases de données
- Systèmes distribués
- Théorie des langages de programmation
- Vérification formelle

Une application : les crypto-monnaies

Critique des systèmes bancaires

- Centralisation : une seule autorité
- Système en boîte noire
- Pas de garantie de fiabilité

Avantages de la blockchain

- Décentralisation : chacun peut participer au système
- Transparence : tout moment est consultable et l'historique est préservé
- Sécurité : chaque participant valide localement les changements et peut rejouer l'état

Crypto-monnaies

Problèmes ouverts

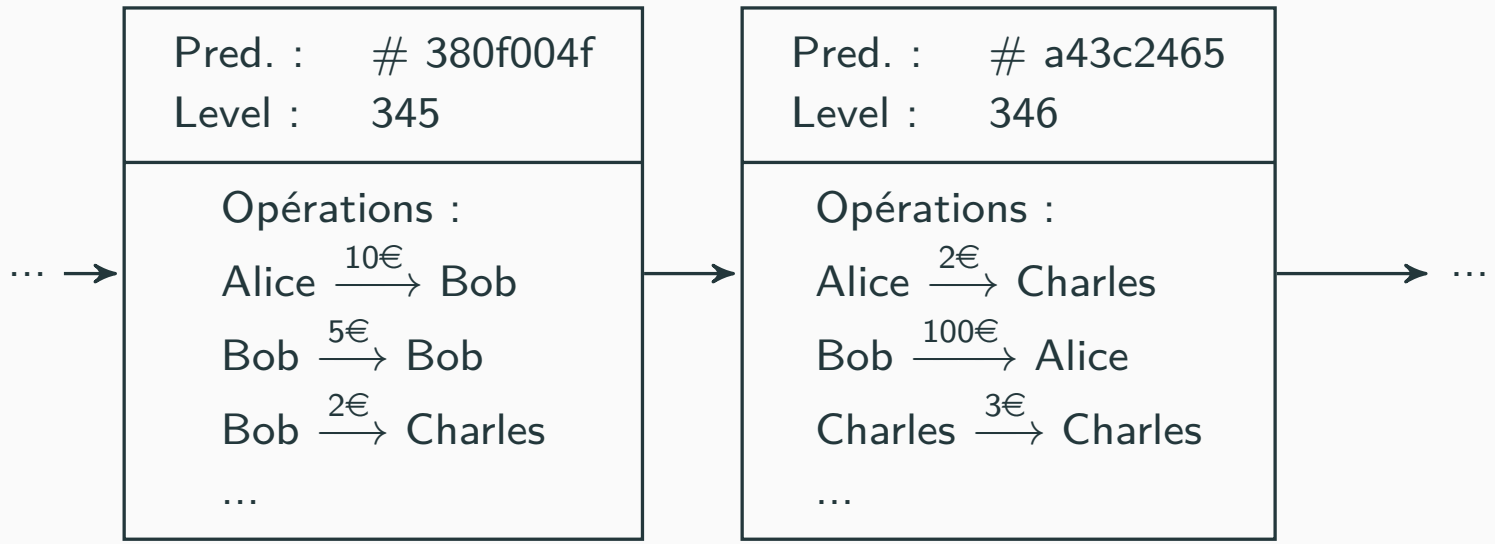
- *Privacy* :
  - L'historique de chaque personne est accessible
  - On veut pouvoir cacher les transactions
- Passage à l'échelle :
  - Conservation de l'entièreté de l'historique
  - Utilisation intensive (spam d'opérations)
- Dissensions : tout le monde peut ne pas être d'accord sur comment devrait fonctionner la chaîne

Quelques projets blockchains

- Bitcoin : 2009 $\rightarrow$ (+ Bitcoin Cash, ...
- Ethereum : 2014 $\rightarrow$ (+ Ethereum Classic, ...)
- Zcash : 2016
- Tezos : 2018

Structure de base de la blockchain

Agrégats d'opérations ordonnés



État de la blockchain

- L'état d'une blockchain est stocké dans une base de données
- La base de données est modifiée après chaque application de bloc
- L'application d'un bloc revient à appliquer l'ensemble des opérations contenues

Application d'un bloc

$\mathcal{B}$

Pred. :	# 380f004f
Level :	345

Opérations :

Alice $\xrightarrow{10€}$ Bob

Bob $\xrightarrow{5€}$ Bob

Bob $\xrightarrow{2€}$ Charles

$\mathcal{S}$

Alice	1230€
Bob	5432€
Charles	543€

$\mathcal{B}(\mathcal{S})$

Alice	1220€
Bob	5440€
Charles	545€

Il faut vérifier ce que l'on reçoit !

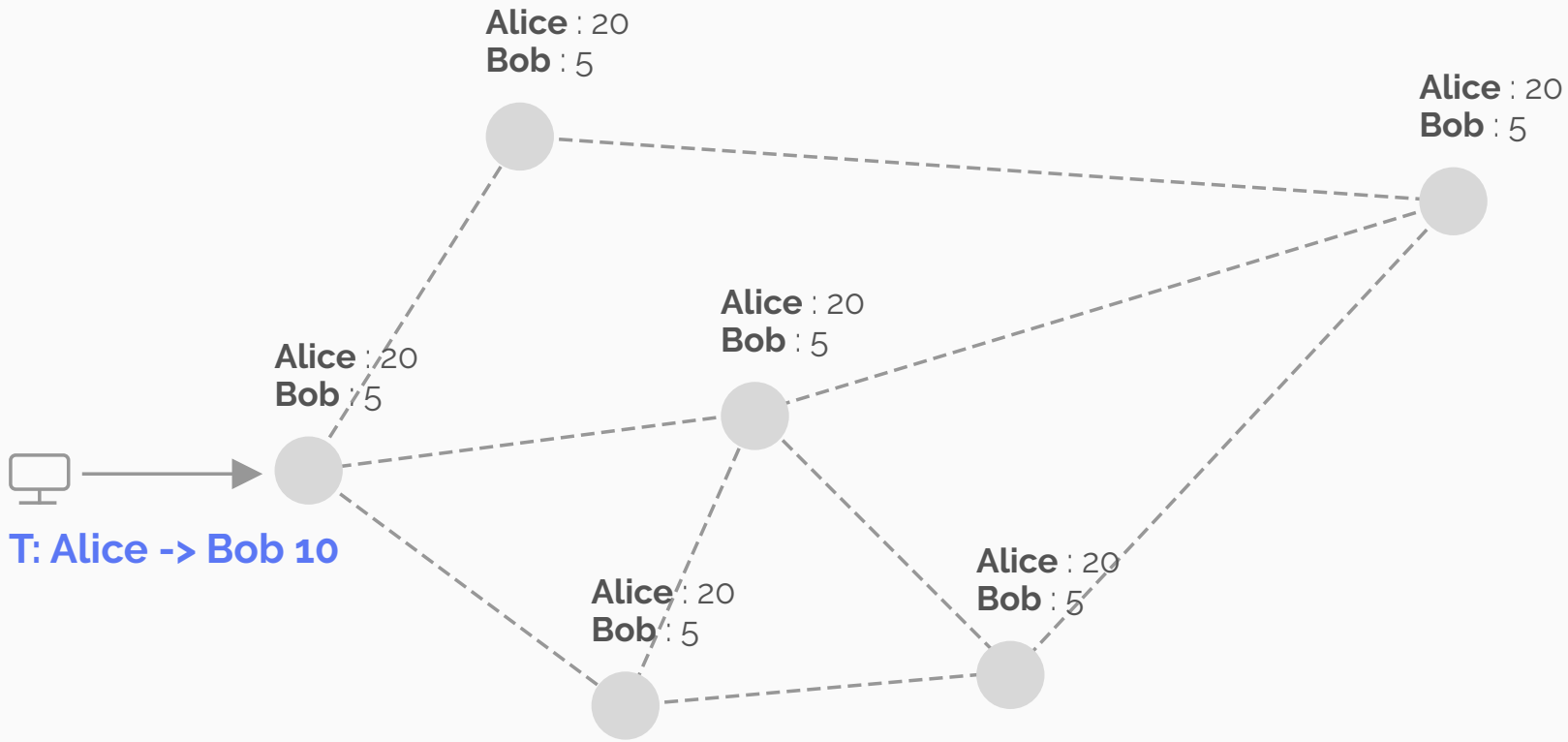
Base de donnée répliquée

- Chaque utilisateur possède une copie locale de l'état de la blockchain
- Cela permet de vérifier localement ce que les autres utilisateurs envoient
- Cela permet également de créer des transactions cohérentes avec l'état actuel de la chaine

Réseau pair-à-pair

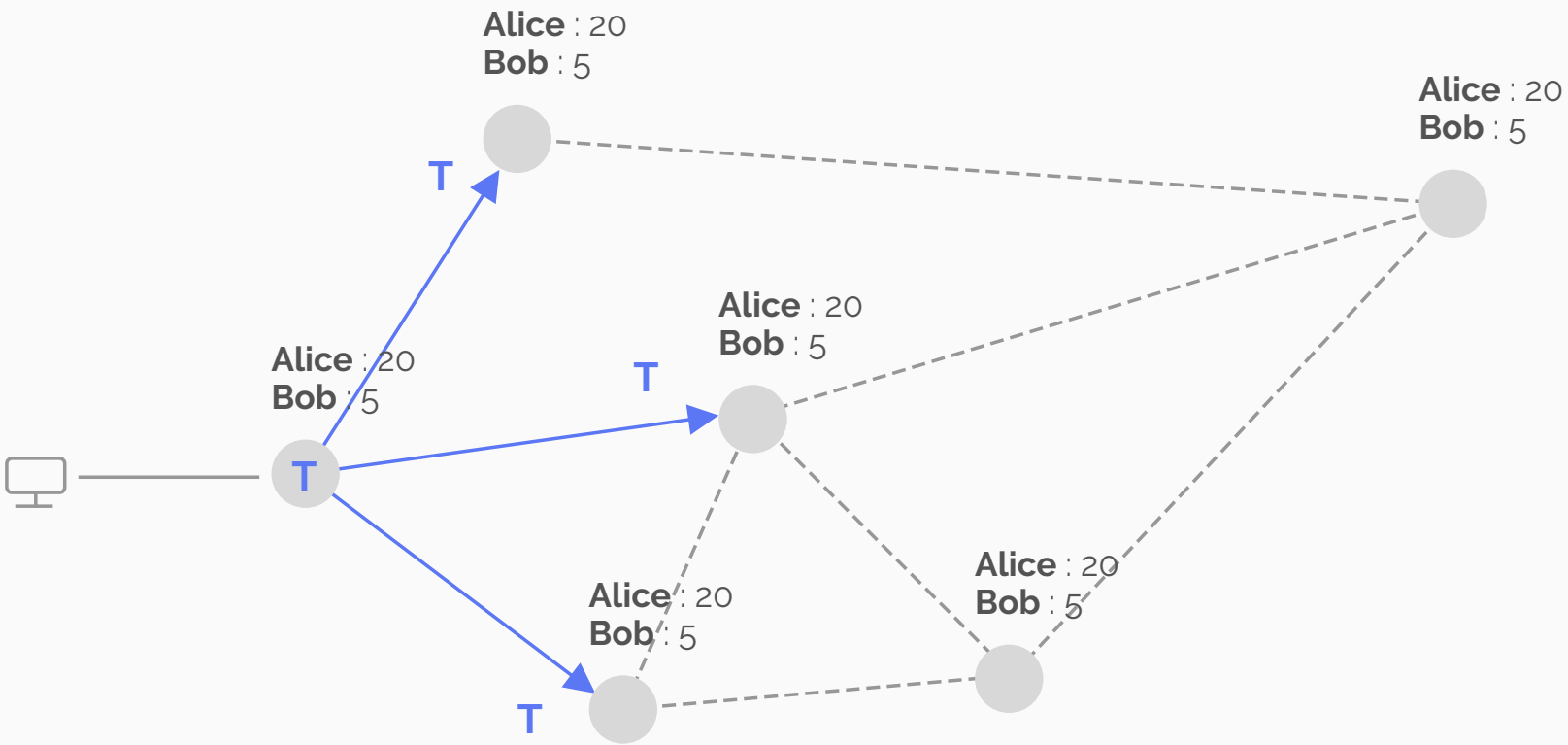
- Les utilisateurs doivent rester synchronisés entre eux
- Ils communiquent via un réseau pair-à-pair en utilisant un **nœud**
- Chaque participant se connecte à un nœud agissant comme un client pair-à-pair
- Plus il y a de nœuds dans le réseau, plus le réseau est sûr

Fonctionnement général



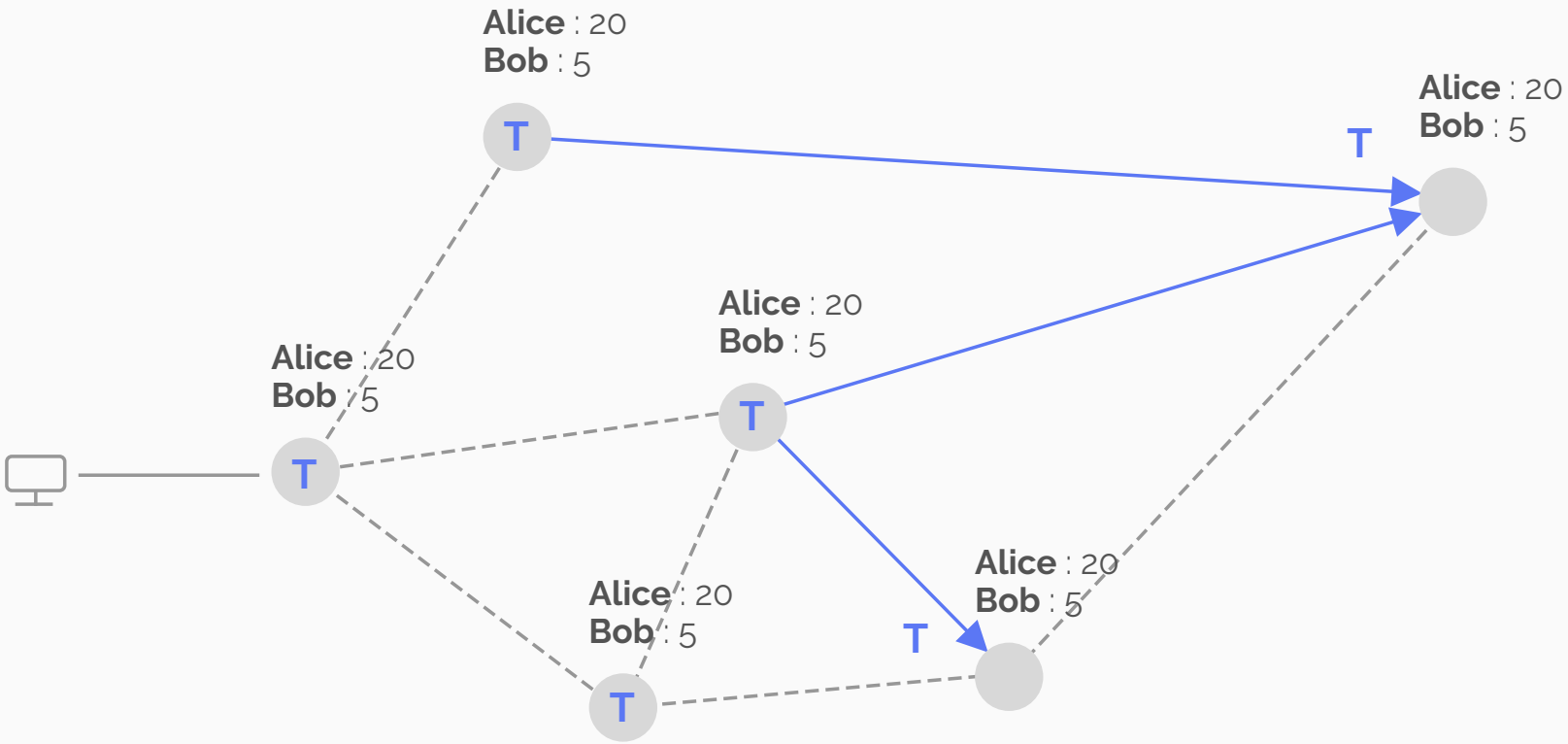
L'utilisateur crée et envoie une transaction à son nœud

Fonctionnement général



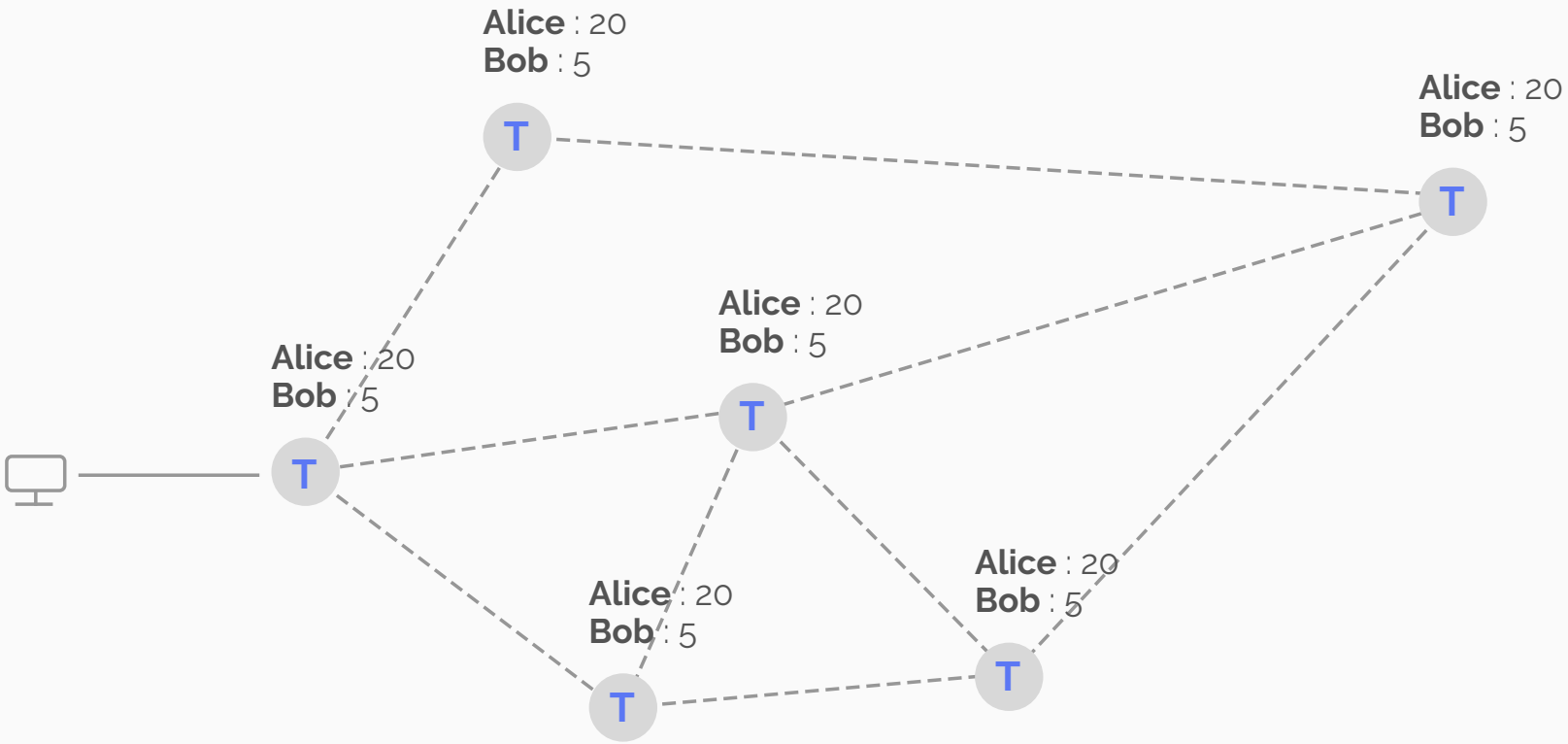
Le nœud propage la transaction à ses pairs qui la mette dans leur liste de transactions

Fonctionnement général



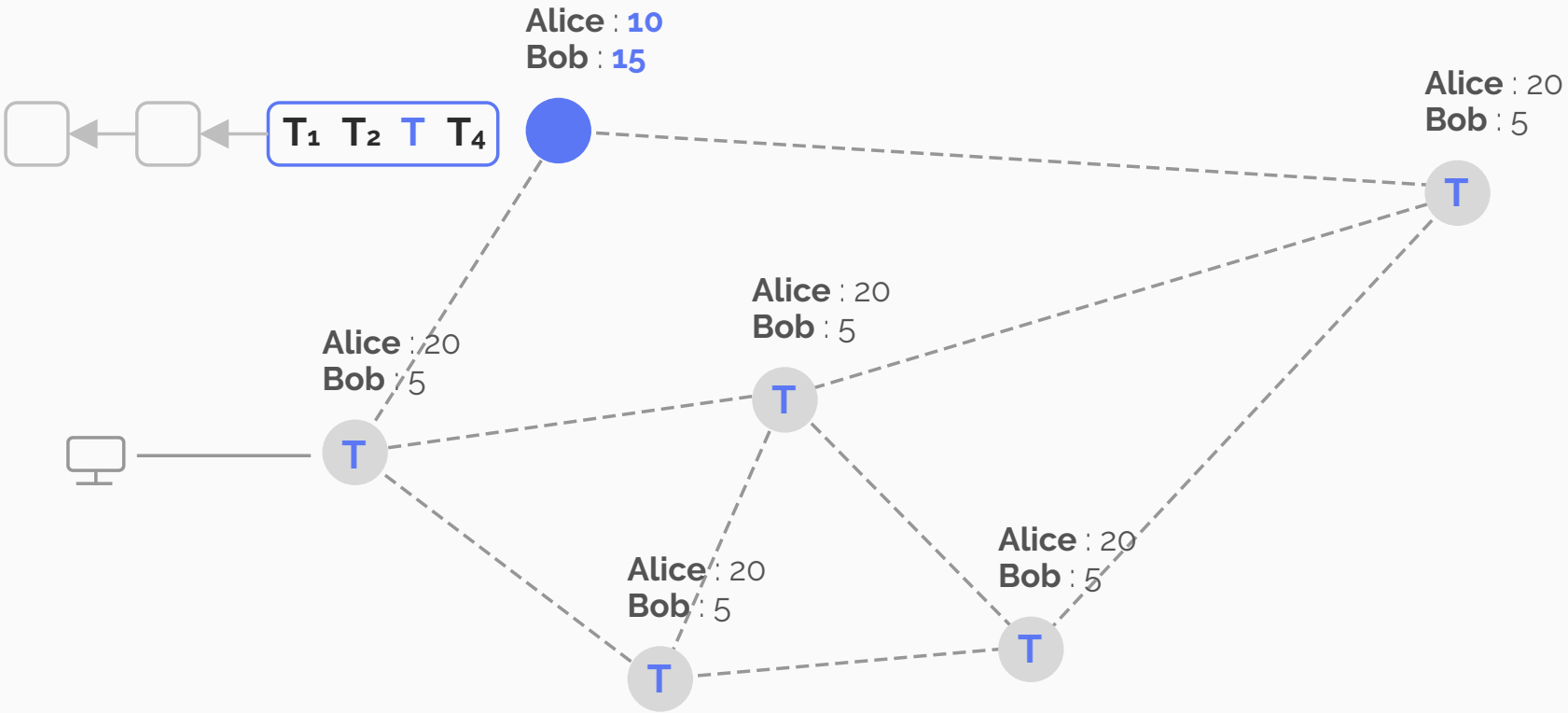
Les autres nœuds informent à leur tour leur pairs...

Fonctionnement général



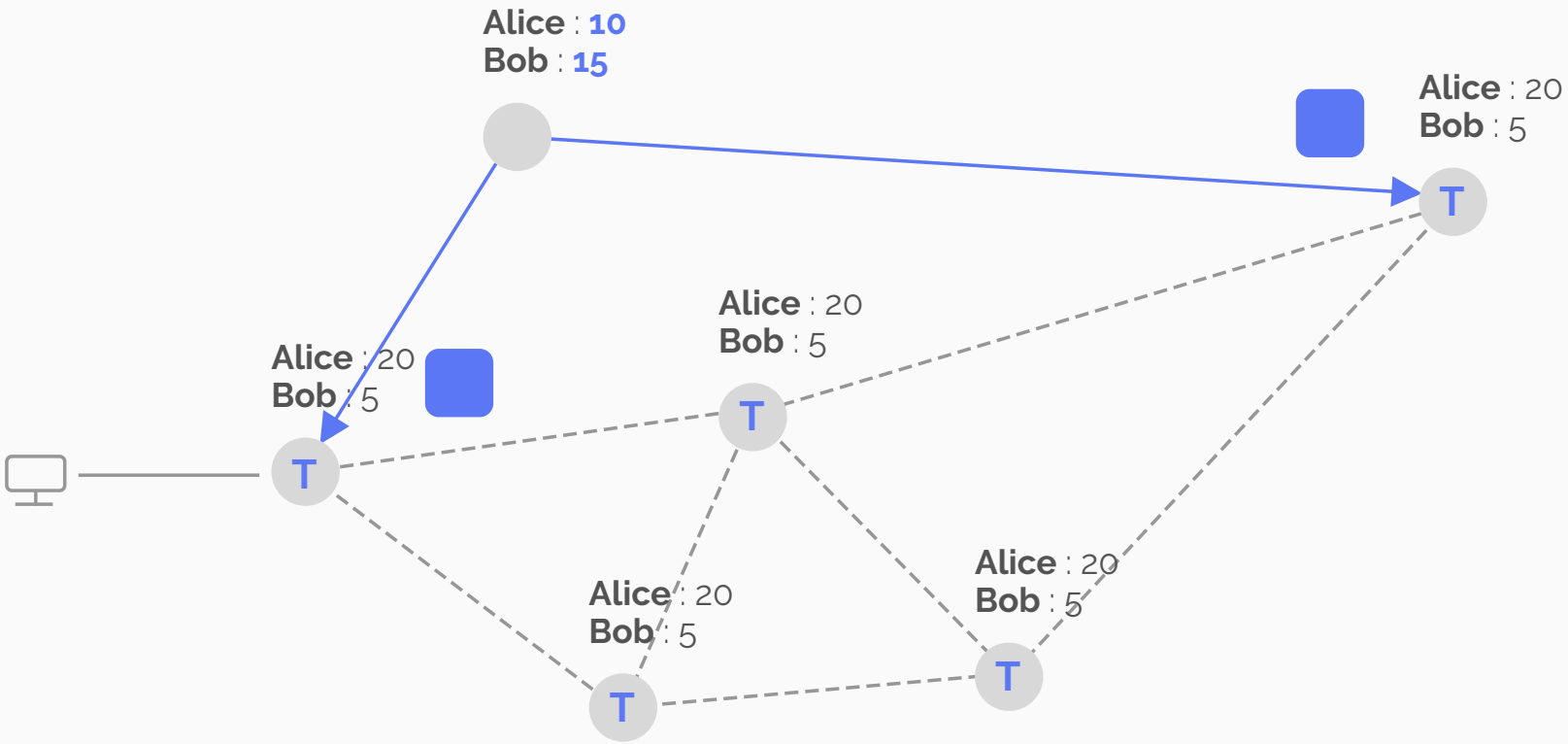
... jusqu'à la propagation globale dans le réseau

Fonctionnement général



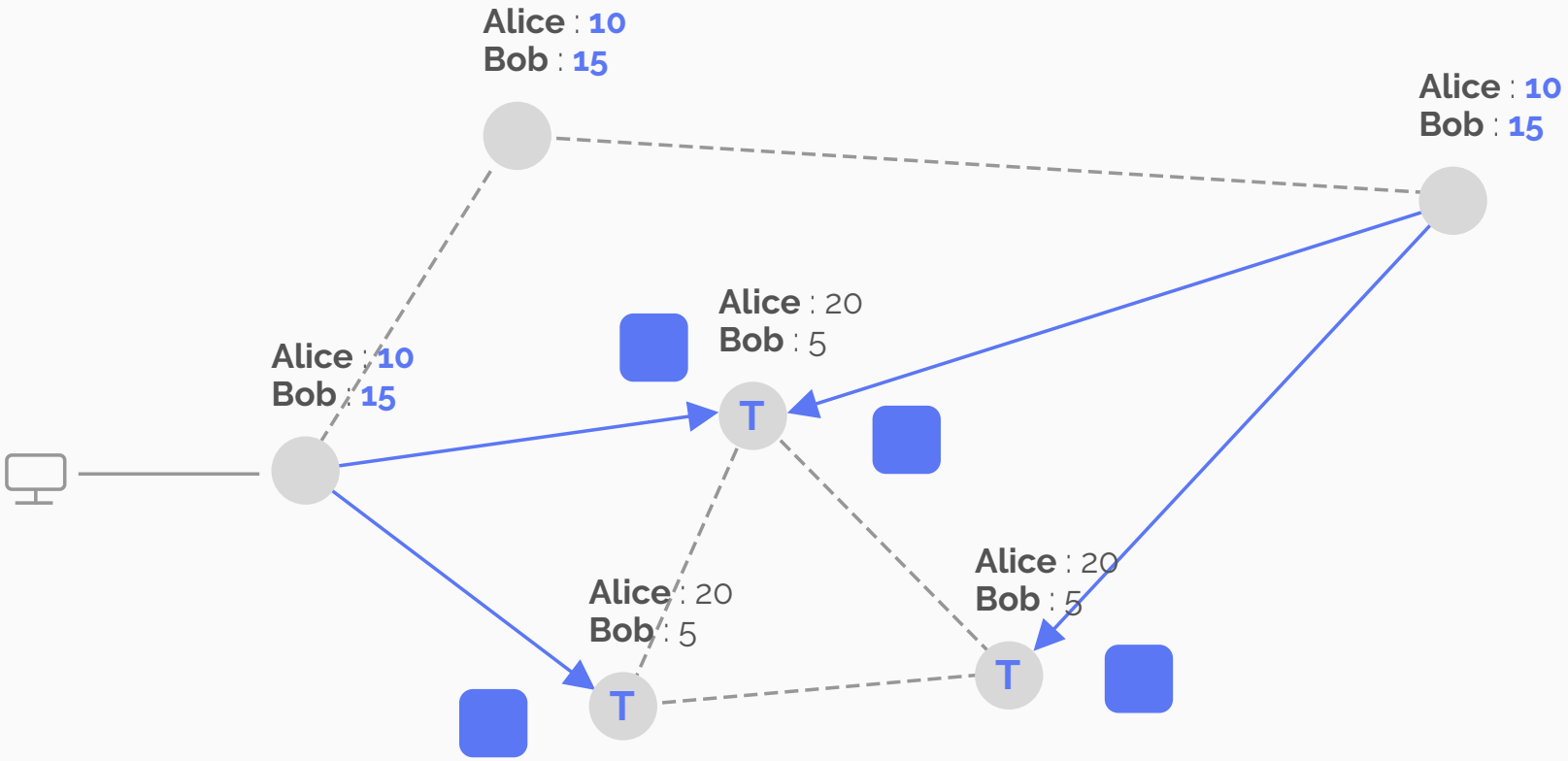
À un moment, un nœud va décider de créer un bloc avec les transactions reçues

Fonctionnement général



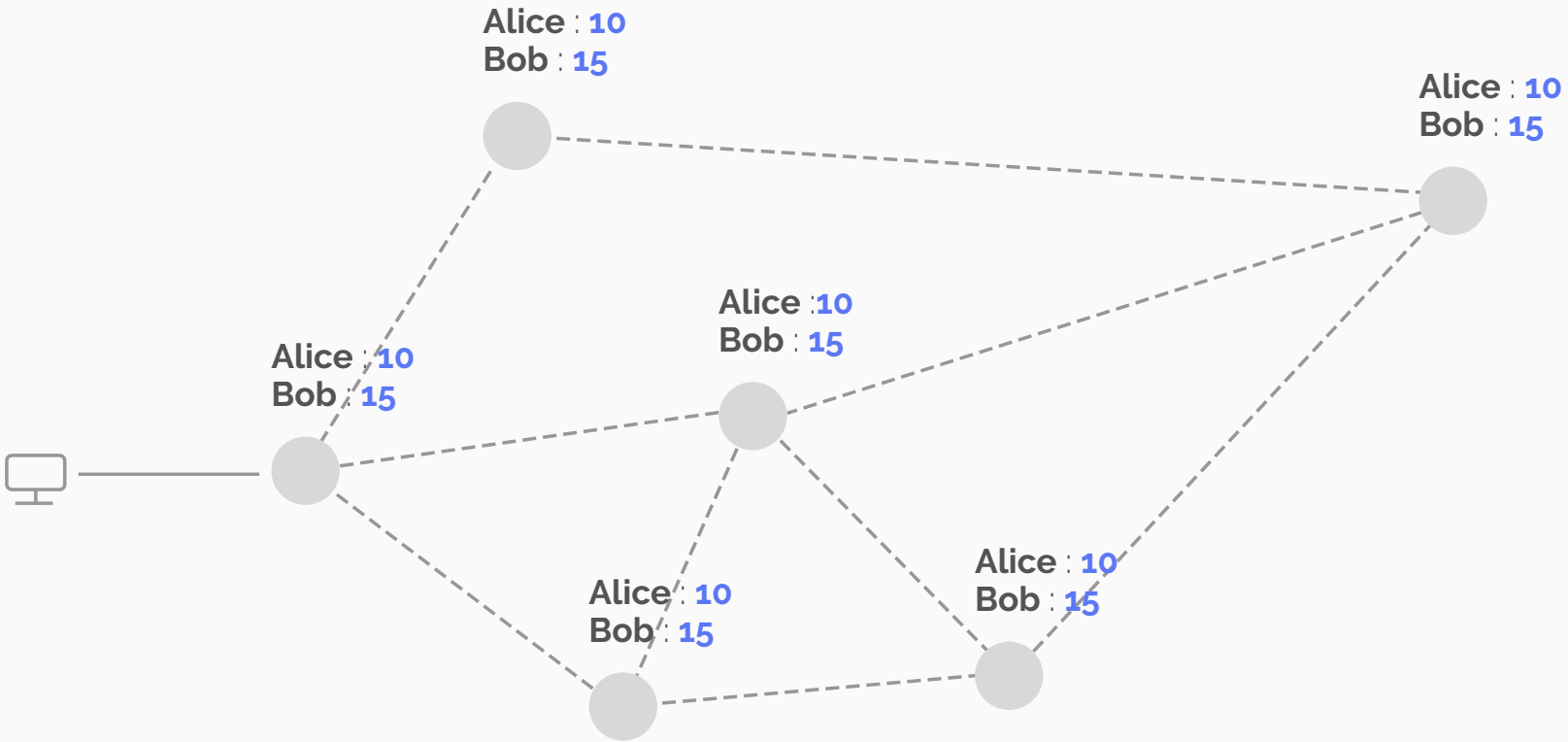
Ce nouveau bloc est propagé et appliqué dans le réseau

Fonctionnement général



Chaque participant vérifie et applique localement ce bloc à sa base de données

Fonctionnement général



Au bout d'un moment, chaque participant a reçu le bloc et la majorité du réseau est à jour

Architecture du système

Une blockchain a besoin de deux composants principaux :

- Un **nœud**
- Un **protocole économique**

Caractéristiques du nœud

Une couche réseau et un stockage

- Couche réseau P2P : *Gossip*
  - Propagation des opérations
  - Propagation des blocs
  - Découverte de pairs réseau
- Stockage :
  - Stocke les blocs et les opérations pour les envoyer au besoin aux autres participants
  - Maintient les états passés de la chaîne pour être capable de revenir en arrière

Protocole économique

Le protocole économique est l'ensemble des règles de la chaîne

Chaque participant accepte de suivre ses règles pour maintenir le bon comportement du réseau

Il détermine notamment :

- la validité des transactions
- la validité des blocs
- attribue un score aux blocs appliqués
- les récompenses à la participation
- ...

Interaction entre le nœud et le protocole

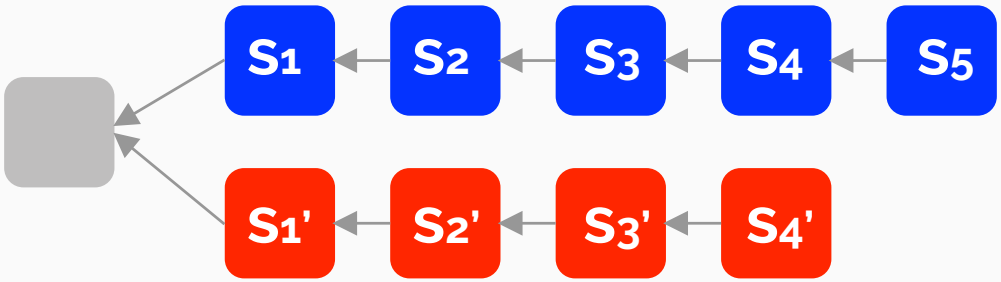
- Le protocole valide les blocs et les opérations
- Le nœud stocke les blocs et les états de la chaîne

Scénario : un nouveau bloc arrive du réseau

1. Le nœud demande au protocole de valider ce bloc ;
2. Le protocole valide le bloc et retourne un score ou si le bloc est invalide rejette le bloc et *kick* au besoin le pair ;
3. Le nœud teste si le nouveau score est meilleur que le score du bloc résultant de son état courant :
   - Si oui : le bloc est sélectionné comme nouvelle tête de chaîne et propagé dans le réseau,
   - Sinon : le bloc est ignoré.

Problème des chaînes concurrentes

Il est possible qu'à un moment le réseau ne soit plus d'accord sur l'état de la chaîne : quel bloc est en tête ?



Pour résoudre ce problème, il est nécessaire d'établir un **algorithme de consensus** que tous les participants sont encouragés à appliquer pour le bon fonctionnement du réseau.

Algorithme de consensus

Un algorithme de consensus doit définir deux notions clés :

- Établir un score à chaque bloc
- Déterminer la validité d'un bloc

Mais pourquoi avoir envie de publier des blocs ?

Incitation économique : chaque personne participant au bon fonctionnement de la chaîne (vivacité) reçoit une récompense

Proof-of-Work

La majorité des implémentations de blockchains actuelles dispose d'un algorithme de consensus basé sur le **proof-of-work**

- Le premier qui arrive à résoudre un puzzle "cryptographique" a le droit de publier un bloc
- Bonnes propriétés : dur de tricher, facile à vérifier, ...

Fonctions de hachage cryptographique

Une fonction de hachage h prend une **donnée** en entrée de taille arbitraire n et produit une image de taille constante c en sortie : un **hash**

$$h : \{0,1\}^n \mapsto \{0,1\}^c$$

Propriétés attendues :

- Déterminisme : $x = y \Rightarrow h(x) = h(y)$
- Uniformité : $x \neq y \Rightarrow h(x) \neq h(y)$ (impossible en pratique)
- Non-inversibilité : pour un hash donné, aucune information sur l'entrée ne doit pouvoir être déduite (cryptographie)

Fonction de hachage cryptographique (2)

Il existe un grand nombre d'algorithmes de hachage cryptographiques

Secure Hash Algorithm (SHA)

```
$ echo "bla" | sha256sum
00e3261a6e0d79c329445acd540fb2b0...

$ echo "bli" | sha256sum
5f2c0653c7f703abf9ac2b083c256a7f...
```

Bitcoin – Algorithme de consensus

Dans Bitcoin, un bloc est dit valide si les bits de son hash commencent par n zéro (**difficulté**)

Block $\mathcal{B}$

Opérations :

Alice $\xrightarrow{1\text{bc}}$ Bob

Roger $\xrightarrow{5\text{bc}}$ John

$h(\mathcal{B}) = 00101011101010 1...$

Si $n = 2$ alors valide

Si $n = 5$ alors invalide

Comment peut-on rendre ce bloc valide ?

Bitcoin – Algorithme de consensus (2)

On ajoute au bloc une valeur (**nonce**) facilement modifiable

Block $\mathcal{B}$

Nonce : <entier>

Opérations :

Alice $\xrightarrow{1bc}$ Bob

Roger $\xrightarrow{5bc}$ John

$\mathcal{B}_i$ = Bloc $\mathcal{B}$ avec i le *nonce* :

- $h(\mathcal{B}_0) = 10101011... \Rightarrow \times$
- $h(\mathcal{B}_1) = 00011101... \Rightarrow \times$
- ...
- $h(\mathcal{B}_{23}) = 0000010... \Rightarrow \checkmark$

Bitcoin – Algorithme de consensus (2)

On ajoute au bloc une valeur (**nonce**) facilement modifiable

Block $\mathcal{B}$

Nonce : <entier>

Opérations :

Alice $\xrightarrow{1bc}$ Bob

Roger $\xrightarrow{5bc}$ John

$\mathcal{B}_i$ = Bloc $\mathcal{B}$ avec i le *nonce* :

- $h(\mathcal{B}_0) = 10101011... \Rightarrow \times$
- $h(\mathcal{B}_1) = 00011101... \Rightarrow \times$
- ...
- $h(\mathcal{B}_{23}) = 0000010... \Rightarrow \checkmark$

Plus n est grand, plus il est "difficile" de trouver un bloc valide

Bitcoin – Algorithme de consensus (3)

Bitcoin vise à ce que le réseau dispose d'un bloc toutes les 10 minutes. Pour arriver à ce but, la difficulté est adaptée dynamiquement :

- Après la publication d'un bloc, la difficulté pour le bloc suivant démarre très haut et décroit chaque minute ;
- Si un bloc valide est trouvé en dessous de 10 minutes, la difficulté sera accrue pour les blocs futurs ;
- Au contraire, si un bloc est publié au-delà de 10 minutes, la difficulté sera réduite.

Bitcoin – Algorithme de consensus (4)

Critiques

- Course au calcul : premier arrivé, premier servi
- Favorise les personnes disposant d'un matériel dédié
- Consomme énormément d'énergie
- Centralise le réseau

Quelques chiffres

- 1 Bitcoin $\approx$ 10.000\$
- La récompense pour un bloc est actuellement 12,5 Bitcoin

Différents algorithmes de consensus

Proof of Work : coûteux mais simple et robuste

D'autres algorithmes de consensus existent :

- Proof of Stake : droit de créer un bloc selon l'investissement des utilisateurs
- Proof of Space : similaire au PoW mais espace disque plutôt que calcul
- Proof of Authority : droit de créer un bloc basé sur la réputation

Conclusion

Thèmes abordés

- Idées générales de la blockchain
- Architecture globale
- Intuition de l'algorithme de consensus du Bitcoin

Thèmes futurs

Cryptographie

- Algorithmes de signature
- Structures cryptographiques
- Transactions anonymes

Consensus

- Détail des principaux algorithmes de consensus
- Nature des attaques potentielles

Thèmes futurs (2)

Privacy

- Sociétalement, la transparence d'une telle structure pose problème
- Comment préserver la vie privée dans un tel registre

Systèmes distribués

- Intéraction entre les différents acteurs du système
- Architecture d'un nœud
- Résistance aux attaques

Thèmes futurs (3)

Protocoles économiques

- Architecture
- Smart-contracts
- Amendements par le vote de protocoles
- Exemple : Protocole économique de Tezos