À la recherche de la meilleure connexion

Analyse des données BGP, RIPE Atlas et PeeringDB

Compilé : 10 avril 2026

Vous travaillez pour *xkcd Incorporated*, une entreprise mondiale implantée dans deux villes. Une nouvelle filiale s'apprête à ouvrir dans une troisième ville. L'entreprise profite de cette extension pour remettre en cause la connectivité sur l'ensemble des trois sites : l'objectif est de garantir une connectivité IPv4 fiable, à faible latence et à haute capacité entre ces trois sites stratégiques. Le défi ? L'équipe d'opérateurs réseau — vous — dispose de trois semaines pour déterminer quels fournisseurs offriraient les meilleures performances à xkcd Inc, pour chacun des sites.

Les trois villes qui vous sont attribuées dépendent de l'option assignée à votre groupe. Cette assignation est disponible sur Moodle. Les options sont les suivantes :

Option	**Ville 1**	**Ville 2**	**Ville 3**
A	Johannesburg	Singapore	Bucharest
B	Auckland	Sydney	Dubai
C	Oslo	Genève	Vienne

TABLE 1 – Options de villes assignées aux groupes

1 Jeux de données

Pour résoudre ce problème, vous utiliserez uniquement les jeux de données disponibles à l'adresse suivante :

Répertoire des jeux de données.

Le répertoire est composé de trois sous-répertoires contenant une extraction des données PeeringDB au 1 janvier 2026, des sauvegardes des tables BGP (RIB), des traceroutes collectés par RIPE Atlas, ainsi que d'autres fichiers utilitaires permettant de vous guider dans ces ressources, présentés ci-dessous.

1.1 Extrait de PeeringDB

PeeringDB est une organisation à but non lucratif qui propose une base de données collaborative recensant les réseaux Internet, les points d'échange auxquels ils sont connectés, ainsi que les centres de données qui les hébergent. L'organisation CAIDA maintient une archive des sauvegardes quotidiennes de ces données (CAIDA peeringDB archive). Le fichier qui vous est fourni en est issu : il s'agit d'un fichier JSON dont la structure est la suivante :

```
{'api': {'data': [...], 'meta': {}}},           % API data
{'as_set': {'data': [...], 'meta': {}}},        % AS sets
{'campus': {'data': [...], 'meta': {...}}},     % campus info
{'carrier': {'data': [...], 'meta': {...}}},    % carriers
{'carrierfac': {'data': [...], 'meta': {...}}}, % carrier facilities
{'fac': {'data': [...], 'meta': {...}}},        % facilities
{'ix': {'data': [...], 'meta': {...}}},         % Internet Exchanges (IXPs)
{'ixfac': {'data': [...], 'meta': {...}}},      % IXP facilities
{'ixlan': {'data': [...], 'meta': {...}}},      % IXP LANs
{'ixpfx': {'data': [...], 'meta': {...}}},      % IXP prefixes
{'net': {'data': [...], 'meta': {...}}},        % networks/ASN
{'netfac': {'data': [...], 'meta': {...}}},     % network facilities
{'netixlan': {'data': [...], 'meta': {...}}},   % network IXP links
{'org': {'data': [...], 'meta': {...}}},        % organizations
{'poc': {'data': [...], 'meta': {...}}}         % points of contact
```

Dans le champ `data` de chaque sous-dictionnaire, vous trouverez une liste de dictionnaires, chacun représentant une entrée de la base de données PeeringDB. Chaque entrée contient des champs spécifiques à son type. Parmi ceux-ci, certains sont des `Clés primaires` servant à identifier de manière unique une entrée, tandis que d'autre sont des `Clés étrangères` permettant d'établir des relations avec des entrées d'autres sous-dictionnaires. Par exemple, le champ `id` dans le sous-dictionnaire `net` est une clé primaire qui identifie de manière unique une entrée, tandis que le champ `org_id` fait référence à une entrée du sous-dictionnaire `org`. Pour trouver à quelle organisation appartient un AS à partir de son entrée dans `net`, il faut trouver l'entrée dans `org` dont le champ `id` correspond au champ `org_id` du sous-dictionnaire `net`.

Le fichier `peeringdb_schema.pdf` présente la liste les champs disponibles pour chaque type d'entrée. Les liens entre les clés primaires et étrangères sont indiqués par des couleurs communes. Une brève description des différentes tables peeringDB est disponible dans le fichier l'extrait intitulé `extract-20220225-Apricot-1-2-GUI-API-with-exercises.pdf`.

À vous d'explorer les données et d'identifier les sous-dictionnaires d'intérêt afin de déterminer les ASs présents dans les villes où votre entreprise est localisée.

1.2 BGP RIBs

Après avoir identifié les AS d'intérêt en utilisant peeringDB, vous vous focaliserez sur les extraits de tables BGP (BGP RIB dumps). Pour cela, vous utiliserez les données fournies par les *collecteurs de routes (RRC)* du service *RIPE Routing Information Service (RIPE RIS)*. Un collecteur est un routeur BGP particulier qui a pour but de fournir un point d'observation sur l'état BGP (chemins et messages reçus) à un endroit du réseau. Pour ce faire, le RRC établit des sessions BGP avec d'autres routeurs (appelés *Vantage Points (VPs)*) qui acceptent d'envoyer toutes les informations qu'ils reçoivent au RRC. Ces interactions sont directement illustrées sur la figure 1. Les données reçues par un RRC sont ensuite typiquement archivées dans le format MRT. L'analyse des chemins BGP sera réalisées en plusieurs étapes, décrites dans le reste de cette section

1.2.1 Identification des VPs d'intérêt

Pour vous éviter de devoir télécharger les dumps BGP de chaque RRC, nous vous fournissons dans ce projet plusieurs fichiers :

— *vps.txt* : liste de tous les VPs présents dans les dumps BGP de RIPE RIS, annotés avec la localisation du VP. Cette liste a été extraite à partir de bgproutes.io

— *vp-to-collector.txt* : association entre chaque VP et la liste des RRC auxquels ce VP est connecté

Sur base de ces informations et des sites où votre entreprise seront présents, nous vous demandons d'identifier quels sont les VPs ainsi que les dumps BGP des RRC d'intérêt.

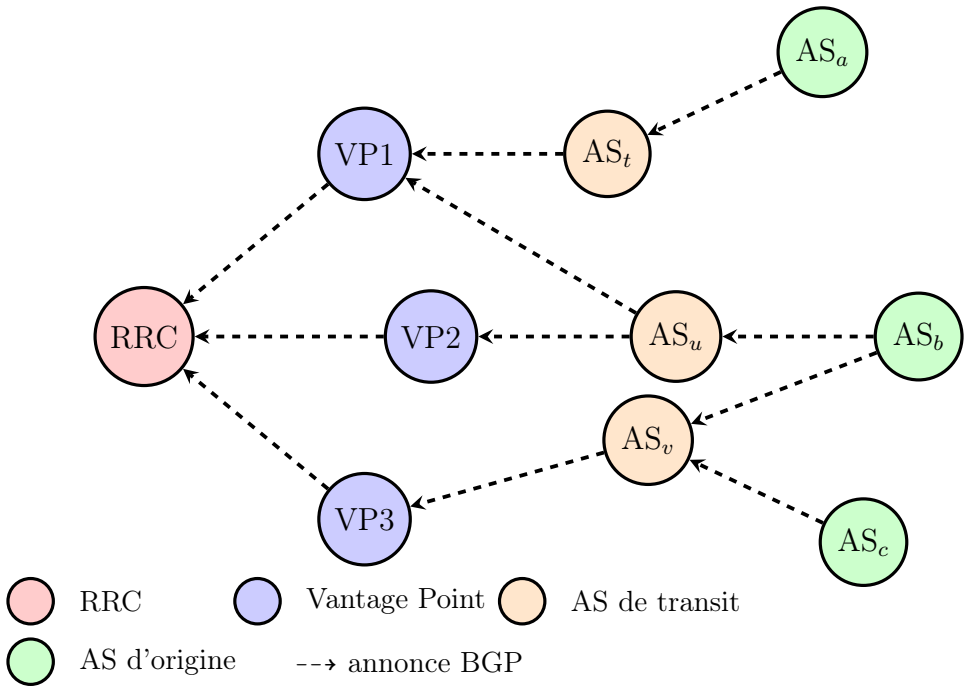

FIGURE 1 – BGP Route Collectors (RRC), Vantage Points (VP), AS de transit et AS d'origine

1.2.2 Inférence des chemins géographiques

Une fois les VPs d'intérêts identifiés à l'étape précédente, nous vous demandons d'extraire les chemins BGP entre ceux-ci. Les chemins vous permettront notamment d'identifier s'il y a des détours géographiques entre un VP et un AS origine. L'objectif est de trouver des chemins qui soient le plus directs possibles. Pour ce faire, il est nécessaire d'utiliser les données de localisation des points de présences fournies par peeringDB. Néanmoins, déterminer quel est le chemin géographique suivi par les paquets à partir du chemin BGP (exprimé en terme d'ASs) nécessite d'inférer la manière dont les ASs sont connectés entre eux. Pour ce projet, vous utiliserez une méthode d'inférence très simple, qui suppose que chaque AS est connecté à l'infrastructure de son voisin la plus proche. D'autres techniques d'inférence plus poussées peuvent également être proposées par vos soins, pour autant que la méthode que vous proposez soit clairement justifiée.

Pour faciliter l'explication de cette technique, nous l'appliquons sur l'exemple de la figure 2. Supposons le chemin `AS1:AS2:AS3:AS4`. Étant donné que, dans ce projet, nous cherchons le chemin entre deux villes, les localisations de AS1 et AS4 sont déjà connues. À la première itération, nous cherchons parmi les infrastructures appartenant à AS2 ($L_{2,1}$, $L_{2,2}$ et $L_{2,3}$), laquelle est la plus proche de AS1. Sur base des distances géographique entre chaque AS1 et chaque infrastructure, nous voyons que $L_{2,2}$ est la plus proche de AS1, et nous supposons donc que le paquet sera envoyé par AS1 à l'infrastructure $L_{2,2}$. Nous répétons ensuite cette opération à partir de $L_{2,2}$, où nous trouvons que l'infrastructure la plus proche de AS3 est $L_{3,1}$. Enfin, nous terminons le chemin en prenant la distance géographique entre $L_{3,1}$ et AS4.

Après avoir utilisé cette technique, comparez la longueur géographique du chemin avec la distance géographique à vol d'oiseau.

1.2.3 Outils

Pour mener à bien ces tâches, nous vous recommandons les outils suivants :

— *bgpkit-parser-py* : module python permettant de parser les fichiers MRT

— *geopy* : module python permettant de calculer des distances géographiques sur base de latitudes et longitudes

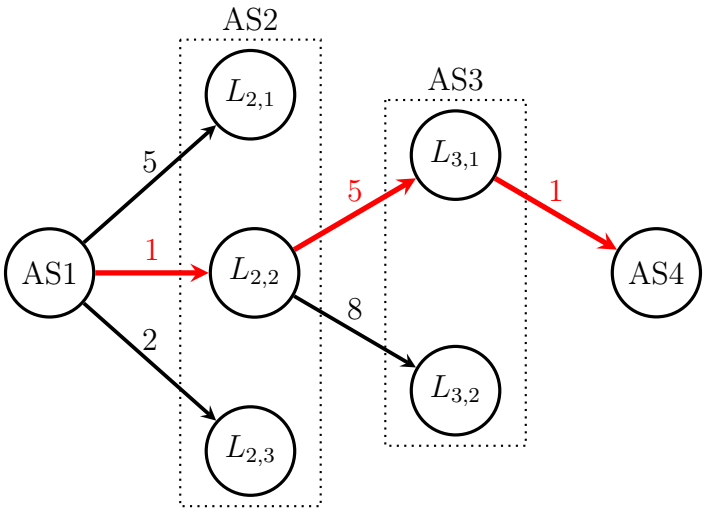

FIGURE 2 – Technique d'inférence géographique appliquée au chemin `AS1:AS2:AS3:AS4`. Le chemin inféré par la technique est mis en évidence en rouge.

1.3 Traceroutes RIPE Atlas

La plateforme RIPE Atlas permet de collecter des données réseau à travers le monde en utilisant des sondes (probes/anchors) installées à différents endroits. Les informations concernant les sondes sont disponibles dans le fichier `anchors_pretty.json` (récupéré directement à partir de l'API RIPE Atlas). Ce fichier contient un tableau avec les informations de chaque probe. Les informations des probes ont le format suivant :

```
{
    "id": ...,
    "as_v4": ...,
    "ip_v4": "...",
    "as_v6": ...,
    "ip_v6": "...",
    "country": "...",
    "city": "..."
}
```

Ces sondes sont utilisées pour effectuer des mesures régulières, notamment des traceroutes. Une sélection de traceroutes vers des destinations d'intérêt pour vos analyses sont fournies dans le dossier `ripe_traceroutes`. Chaque fichier est nommé selon le format `traceroutes_x_x_x.json`, où `xxx` correspond aux codes des pays d'où les mesures ont été effectuées. Chaque fichier contient un tableau de mesures traceroute qui ont le format suivant :

```
{
    "af": 4,
    "dst_addr": "...",
    "dst_name": "...",
    "endtime": ...,
    "from": "...",
    "fw": ...,
    "group_id": ...,
    "lts": ...,
    "msm_id": ...,
    "msm_name": "Traceroute",
```

```
    "paris_id": ...,
    "prb_id": ...,
    "proto": "...",
    "result": [:results],
    "size": ...,
    "src_addr": "...",
    "stored_timestamp": ...,
    "timestamp": ...,
    "type": "traceroute"
  }
```

Chaque élément de `results` correspond à un traceroute entier, lui-même composé de plusieurs sauts. Chaque saut est représenté par un dictionnaire contenant les champs suivants :

```
    "hop": ...,
    "result": [
        {
        "from": "...",
        "ttl": ...,
        "size": ...,
        "rtt": ...,
        ("icmpext": ...)
        },
        ...
    ]
```

ou

```
    "hop": ...,
    "result": [
        {
            "x": "*"
        },
        ...
    ]
```

En fonction des informations disponibles pour chaque saut.

En complément de ces données, le fichier `ip_infos.json` contient un dictionnaire associant à chaque adresse IP présente dans les traceroutes les informations suivantes (venant de https://ipinfo.io/) :

```
{
    "ip": "..",
    "asn": "AS...",
    "as_name": "...",
    "as_domain": "...",
    "country_code": "...",
    "country": "...",
    "continent_code": "..",
    "continent": "..."
}
```

Utiliser ces traceroutes pour déterminer les chemins suivis par les paquets entre les villes où vous êtes présents pour les ASs qui sont vos candidats en tant que fournisseurs de connectivité. Ces traceroutes vous donnent également des informations sur les performances observées ainsi qu'éventuellement les pertes entre paires de noeuds.

2 Attentes

Ce travail est à réaliser par groupe de 3.
Vous rendrez les éléments suivants :

1. Un rapport, sous forme de slides, décrivant votre approche et le choix résultant de fournisseur pour chaque site. Ces slides seront celles que vous utiliserez lors de la présentation, consistant en 6 min de présentation et 4 min de question et réponses. Vos slides seront réalisées en prenant en compte cette limite de temps ;
2. Un lien vers le dépôt **privé** que vous avez créé sur la Forge UCLouvain, contenant l'intégralité de votre code. Votre dépot sera nommé `LINFO1341-PROJECT2-XX`, où `XX` est votre numéro de groupe. Votre code doit être écrit en python, et contient l'entièreté de l'analyse des données effectuée. Il génère en sortie le choix de vos fournisseurs pour chaque ville où *xkcd Incorporated* est présent, la liste des chemins niveau AS entre ces villes, une quantification de la déviation minimale par rapport au chemin à vol d'oiseau, la liste des chemins au niveau IP, le délai min, médian et max entre les villes ainsi qu'une quantification de la qualité des performances vers le reste de l'Internet. Assurez-vous que votre code soit accessible à l'équipe enseignante, dont les identifiants Forge sont repris ci-dessous.
   - @andoeraene
   - @AurelienBuchet
   - @burlats

La date limite pour le dépôt du rapport et de partage d'accès au dépôt Git est fixée au **lundi 4 mai 2026 à 14h00**, via Moodle.

Une soutenance orale aura lieu la semaine du **4 mai 2026** durant laquelle vous expliquerez votre approche, les outils utilisés et les données exploitées tout au long du projet.

L'objectif de cette défense orale est de nous assurer de la bonne compréhension du projet par **tous les membres du groupe**. Les autres modalités de la présentation seront communiquées ultérieurement.

> ☞ Important : une absence injustifiée ou un échec à défendre votre projet oralement entraînera l'échec global du projet.

Aucun partage de solutions entre groupes ne sera toléré. Chaque membre du groupe doit maîtriser l'ensemble des outils et méthodes utilisés, être capable d'analyser en profondeur et de valider rigoureusement les résultats obtenus, et démontrer qu'il peut appliquer les connaissances et compétences acquises pour raisonner de manière autonome sur ce type de problèmes. Nous serons très attentifs à tous ces aspects.