# Extraction probabiliste de formes SHACL à l'aide d'algorithmes évolutionnaires

Rémi Felin, Pierre Monnin, Catherine Faron, Andrea G. B. Tettamanzi

# Extraction probabiliste de formes SHACL à l'aide d'algorithmes évolutionnaires

Rémi Felin, Pierre Monnin, Catherine Faron, Andrea Tettamanzi

Université Côte d'Azur, Inria, I3S, Sophia-Antipolis, France
{prénom.nom}@inria.fr

**Résumé.** SHACL est la recommandation du W3C pour la représentation de contraintes sur des graphes RDF, appelées formes (*shapes* en anglais), utilisées pour valider des données RDF. La création de formes capturant les contraintes du domaine étant une tâche très fastidieuse, des travaux récents abordent le problème de leur génération automatique. Dans ce contexte, nous proposons une approche combinant l'évolution grammaticale et un cadre probabiliste pour générer des formes SHACL candidates et valider des graphes RDF par rapport à celles-ci. L'approche est validée en l'appliquant à l'extraction de formes SHACL représentant des règles d'associations, à partir d'un ensemble de données RDF. Les résultats montrent le potentiel et la généralité de l'approche proposée.

## 1 Introduction

Ces dernières années, le volume des données RDF hétérogènes disponibles a considérablement augmenté, notamment avec de multiples initiatives mondiales de production de données ouvertes liées (Linked Open Data). Celles-ci forment un vaste ensemble de jeu de données RDF interconnectés dans divers domaines dont la masse disponible ouvre à son tour de nouvelles possibilités d'exploitation pour en extraire de nouvelles connaissances et les améliorer. Dans ce papier, nous abordons le problème général de la découverte de contraintes de domaine à partir de données RDF. En 2017, Kontokostas et Knublauch (2017) ont recommandé SHACL comme langage pour exprimer des contraintes sur les graphes RDF sous la forme de modèles de graphes appelés formes (*shapes* en anglais) auxquels les données RDF doivent se conformer dans un certain domaine ou une certaine application, par exemple pour de l'intégration ou correction de données. SHACL permet d'exprimer un large éventail de contraintes sur les nœuds et les propriétés d'un graphe RDF. Néanmoins, le volume croissant et l'hétérogénéité des données RDF disponibles rend la création manuelle de contraintes SHACL difficile, motivant la question de recherche suivante : *comment extraire des formes SHACL à partir d'un graphe de données RDF de manière automatique ?*

Nous abordons cette question en proposant une approche utilisant l'évolution grammaticale, un type particulier d'algorithme évolutionnaire, pour générer des formes SHACL candidates en tenant compte d'une grammaire BNF (Backus-Naur Form) donnée, qui permet l'instanciation d'une population de formes candidates bien définies et destinées à évoluer par le biais d'un processus évolutionnaire. Ce processus est adapté à la tâche car il permet d'explorer le large espace de recherche associé pour trouver des formes valides et diversifiées. Cette

approche exploite également un cadre probabiliste pour la validation SHACL. Ceci permet de tolérer un taux d'erreur physiologique dans les données RDF lors du processus de validation SHACL des formes candidates, ce qui est nécessaire face à l'hétérogénéité et l'incomplétude inhérente aux données RDF libres et ouvertes. L'approche est validée en l'appliquant à l'extraction de formes SHACL à partir d'un ensemble de données RDF relatives au domaine scientifique.

Le reste du document est organisé comme suit : dans la section 2, nous résumons les travaux connexes et positionnons notre travail au sein de cet espace de recherche ; dans la section 3, nous introduisons les éléments préliminaires à notre contribution : (i) cadre probabiliste, (ii) algorithmes évolutionnaires et évolution grammaticale. Dans la section 4, nous présentons notre approche proposée pour l'extraction de formes SHACL à partir de données RDF. Les résultats de nos expériences sont présentés dans la section 5, et nous concluons par une discussion sur les recherches futures dans la section 6.

## 2 Travaux connexes

L'extraction automatique et semi-automatique de contraintes est actuellement un domaine de recherche actif (Rabbani et al., 2022), l'accent étant mis sur l'extraction de formes SHACL et/ou de formes ShEx [1]. Plusieurs approches génèrent des contraintes SHACL à partir d'ontologie OWL (Keely, 2020; Cimmino et al., 2020; Fernández-Álvarez et al., 2018; Saleh et al., 2022). Pandit et al. (2018) proposent de construire des formes SHACL en utilisant un modèle de conception d'ontologie (Ontology Design Pattern) mais ne fournit pas de moyen automatique ou semi-automatique de générer des formes.

Plusieurs approches extraient des formes SHACL ou ShEx des données RDF (Keely, 2020; Boneva et al., 2019; Fernández-Álvarez et al., 2022; Omran et al., 2022). Plus précisément, Boneva et al. (2019) proposent une extraction semi-automatique des formes SHACL exprimant les contraintes de cardinalité et de type de valeur sur les propriétés. Ces formes peuvent être personnalisées par les utilisateurs au moyen d'une interface graphique, et un large ensemble de contraintes peut être extrait pour un ensemble spécifique de données RDF. Cependant, cette approche peut ne pas être adaptée aux grands ensembles de données RDF, comme c'est le cas de celle présentée par Rabbani et al. (2022). Une méthode similaire, proposée par Fernández-Álvarez et al. (2022), aborde le problème de l'extensibilité en optimisant la consommation de mémoire. Mihindukulasooriya et al. (2018) s'appuient quant à eux sur des techniques d'apprentissage automatique pour générer automatiquement des formes RDF en utilisant des données RDF profilées comme caractéristiques. Enfin, Omran et al. (2022) présentent une approche qui exploite les mesures de qualité pour apprendre des règles entre les entités. Ces règles peuvent ensuite être traduites en formes SHACL, tout en tenant compte de l'incertitude inhérente aux grands graphes de connaissances. Cependant, cette approche est limitée à la génération de certaines règles spécifiques.

Par rapport à l'état de l'art, notre travail se concentre sur l'extraction automatique des contraintes SHACL (hors ShEx) à l'aide d'une approche évolutionnaire. La production de formes candidates repose sur une grammaire BNF qui permet l'expression de tous les types de cibles et de contraintes de SHACL *Core* [2]. La validation probabiliste des formes candidates

1. https://shex.io/
2. https://www.w3.org/TR/shacl/#core-components

permet de considérer un taux d'erreur théorique contenu dans les données RDF.

# 3 Éléments préliminaires

## 3.1 Une évaluation SHACL probabiliste

Dans un contexte réel, les ensembles de données RDF sont imparfaits, incomplets avec des erreurs de différentes natures. Ainsi, lors de l'extraction de formes SHACL à partir d'un ensemble de données RDF, les formes candidates pourraient déclencher quelques violations dans les données du fait de la qualité de l'ensemble de données. Pour prendre cela en compte lors de l'extraction de formes SHACL, nous nous appuyons sur le cadre probabiliste pour la validation SHACL proposé par Felin et al. (2023). Il s'agit d'une extension de l'évaluation SHACL des données RDF par rapport aux formes en tenant compte d'une proportion d'erreur théorique physiologique $p$ dans les données et donc d'un taux de violation acceptable possible. Dans le reste de cette section, nous résumons les principes de ce modèle.

Si l'on considère un graphe RDF $\upsilon$, le **support** d'une forme $s$, $\upsilon_s$, est l'ensemble des triplets de $\upsilon$ ciblés par $s$ (et donc testés lors de la validation). La **cardinalité de référence** (refCard) de $s$ est la cardinalité de son support : $|\upsilon_s|$. Les **confirmations** et **violations** de $s$, respectivement $\upsilon_s^+$ et $\upsilon_s^-$, sont les ensembles de triplets qui, respectivement, sont cohérents avec $s$ et violent $s$ : $\upsilon_s^+ \cap \upsilon_s^- = \emptyset$, et $\upsilon_s = \upsilon_s^+ \cup \upsilon_s^-$. Le modèle probabiliste pour la validation SHACL repose sur une distribution binomiale $X \sim B(|\upsilon_s|, p)$ où $p$ est la proportion d'erreurs théoriques inévitables. La **vraisemblance** d'observer un nombre de violations $|\upsilon_s^-|$ dans le support d'une forme $s$ compte tenu de $X \sim B(|\upsilon_s|, p)$ est définie comme suit :

$$L_{|\upsilon_s^-|} = P(X = |\upsilon_s^-|) = \binom{|\upsilon_s|}{|\upsilon_s^-|} \cdot p^{|\upsilon_s^-|} \cdot (1-p)^{|\upsilon_s^+|} \tag{1}$$

L'**acceptation** d'une forme $s$ dépend de la proportion de violations observée pour $s$. Ainsi, $s$ est compatible avec $\upsilon$ si $\hat{p_s} = \frac{|\upsilon_s^-|}{|\upsilon_s|}$ est plus petit que la proportion théorique de violations $p$ :

$$\hat{p_s} \leq p \implies KG \models s. \tag{2}$$

Dans le cas où $\hat{p_s} > p$, un test d'adéquation du khi-deux est effectué à l'aide de la *statistique de test* $X_s^2$ définie comme suit :

$$X_s^2 = \sum_{i=1}^{k} \frac{(n_i - T_i)^2}{T_i} \sim \chi^2_{k-1;\alpha}. \tag{3}$$

où $k$ est le nombre total de groupes, soit $k = 2$, $n_i$ est ici le nombre observé de violations et $T_i$ est le nombre théorique de violations. L'acceptation de l'hypothèse nulle $H_0$ implique l'acceptation de $s$ :

$$X_s^2 \leq \chi^2_{k-1;\alpha} \implies KG \models s \tag{4}$$

## 3.2 Les algorithmes évolutionnaires

Les algorithmes évolutionnaires sont une famille d'algorithmes d'optimisation stochastique basés sur une population, qui simulent les mécanismes de la sélection naturelle pour

résoudre des problèmes complexes. Les *algorithmes génétiques*, la *programmation évolutive*, les *stratégies d'évolution* et la *programmation génétique* sont les quatre sous-domaines qui composent les algorithmes évolutionnaires. Les individus d'une population sont représentés sous forme d'expressions symboliques, par exemple des ensembles de valeurs binaires, qui peuvent être interprétées et exécutées par un système informatique (Baeck et al., 2000).

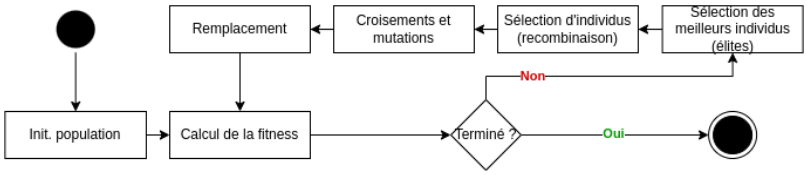


FIG. 1 – *Vue simplifiée du fonctionnement d'un algorithme évolutionnaire.*

La figure 1 présente les étapes fondamentales de l'algorithme. L'objectif premier de ces algorithmes est d'obtenir une bonne solution ou un ensemble de bonnes solutions (*locales* ou *globales*) qui répondent au mieux à la problématique. La première étape consiste à générer une population d'individus de manière aléatoire, où chaque individu représente une solution possible au problème. La fonction de *fitness* est une expression mathématique qui décrit le degré d'aptitude d'un individu en tant que solution au problème. Ce score est utilisé pour comparer les individus et les sélectionner pour la reproduction. Les algorithmes évolutionnaires s'appuient sur différents opérateurs pour converger vers une nouvelle population censée être mieux adaptée au problème considéré à travers un processus évolutif.

On distingue la **sélection *élitiste*** des meilleures individus, qui survivent jusqu'à ce que de meilleures individus soient découverts, et la **sélection *recombinaison*** d'individus pour le processus de recombinaison (ou de reproduction) qui donne une plus grande chance de reproduction aux individus les mieux adaptés de la population. Les individus sélectionnés pour la reproduction subissent le processus de **croisement** (crossover en anglais) et de **mutation** avec une certaine probabilité d'occurrence. Le croisement consiste à échanger un ou plusieurs segments d'information génétique entre deux individus, tandis que la mutation, moins fréquente, entraîne une modification aléatoire d'une ou plusieurs caractéristiques d'un individu. Un individu, issu de la reproduction, est intégré dans la population de remplacement si et seulement s'il n'est pas déjà représenté dans cette même population et représenté dans la population élitistes (afin d'éviter un phénomène de duplication). Enfin, la nouvelle population issue de ce processus vient **remplacer** les individus de la population ne faisant pas parti de la population élitiste. Les nouveaux individus sont par la suite **évalués** afin de déterminer leur score de fitness, ou autrement dit leur aptitude à répondre à la problématique. La condition d'arrêt de l'algorithme est déterminé par **l'effort** investi à la découverte d'un large ensemble de solutions candidates crédibles. Cet effort divisé par la taille de la population indique le nombre de **générations** au cours desquelles la population évoluera.

## 3.3 L'évolution grammaticale

L'évolution grammaticale est un type particulier de programmation génétique qui permet de générer automatiquement des expressions de longueur variable dans n'importe quel langage (O'Neill et Ryan, 2001). Dans notre cas, nous considérons un ensemble d'expressions

```
1 Shape := " a " NodeShape
2 NodeShape := "sh:NodeShape ; " ShapeBody
3 ShapeBody := "sh:targetClass " CLASS " ; " ShapeProp
4 ShapeProp := " sh:property [ " PropBody " ] . "
5 PropBody := " sh:path rdf:type ; sh:hasValue " CLASS " ; "
```

FIG. 2 – *Extrait d'une grammaire BNF.*

représentant une *population* $P$ où chaque individu constitue une forme SHACL candidate. Une *grammaire BNF* fournit la caractérisation *phénotypique* et *génotypique* pour chaque individu à l'aide d'un ensemble de *règles statiques* et de *règles dynamiques*. Les règles statiques sont les composantes *immuables* tandis que les règles dynamiques (Nguyen et Tettamanzi, 2019) sont les composantes *variables* du caractère phénotypique, où chaque règle a une ou plusieurs valeurs possibles et où chaque valeur est identifiée par un phénotype. Des exemples de composantes immuables et variables sont détaillés en section 4.

Un individu est constitué d'un **génotype**, qui est un ensemble de nombres entiers (ou codons) correspondant à l'identifiant de l'information qu'il caractérise, par exemple `[1784, 572, 1002, 14]`. Cet ensemble est exploité par l'algorithme dans le processus d'évolution (crossover, mutation, ...). Chaque codon traduit un morceau d'information, l'ensemble formant un individu lisible par l'humain. L'information complète est exprimée par le **phénotype**, par exemple `<myNodeShape> a sh:NodeShape`.

# 4 Découverte évolutive de formes SHACL

## 4.1 Vue d'ensemble

Nous adoptons une approche d'évolution grammaticale décrite dans la section 3.2 et nous l'appliquons à la découverte des formes SHACL. Nous avons défini des gammaires BNF conformes à la recommandation (Kontokostas et Knublauch, 2017) pour fournir la caractérisation phénotypique des formes SHACL à produire par notre algorithme d'évolution grammaticale. Il peut s'agir d'un fragment précis comme les contraintes sur les types de valeurs [3] ou de l'intégralité des contraintes SHACL. À titre d'exemple, la figure 2 est un extrait de la grammaire BNF permettant de générer des formes SHACL avec une contrainte `sh:hasValue`. `CLASS` est une règle dynamique permettant d'extraire automatiquement toutes les classes possibles de l'ensemble de données RDF à l'aide d'une requête SPARQL. Se basant sur cet exemple, le génotype d'une forme est une paire de codons $(i, j)$, qui sont décodés en deux classes issues de l'ensemble de données RDF et injectées dans la structure phénotypique suivante produite par la grammaire : `"a sh:NodeShape ; sh:targetClass "CLASS[i]" ; sh:property [ sh:path rdf:type ; sh:hasValue "CLASS[j]" ; ] ."` Ici, la mutation correspond à changer l'un ou les deux entiers $i$ et $j$. La recombinaison correspond à croiser $(i, j)$ avec un individu $(i', j')$ pour obtenir, par exemple, $(i, j')$ et $(i', j)$. Notons que les individus sont initialisés aléatoirement au début du processus.

3. https://www.w3.org/TR/shacl/#core-components-value-type

Pour évaluer l'efficacité de notre algorithme évolutionnaire, comme l'évolution de la population au cours des générations successives, nous utilisons les métriques suivantes. L'analyse de l'évolution du **score de fitness moyen d'une population** nous donne une idée très claire sur l'aptitude de notre algorithme à découvrir de solutions toujours plus satisfaisantes à mesure que la population évolue. Un score de fitness moyen convergeant vers une valeur fixe traduit la découverte d'une solution globale tandis qu'un score de fitness moyen en croissance continue traduit la découverte de multiples solutions locales (le cas souhaitable dans ce contexte). Le **coefficient de diversité** est égal au nombre d'individus distincts dans une population donnée à une génération spécifique divisé par la taille totale de la population. Cette mesure est utilisée pour évaluer la diversité globale d'une population, un taux faible indiquant une forte proportion d'individus dupliqués. En règle générale, ce phénomène peut être le signe d'une solution très satisfaisante qui a produit un grand nombre de descendants. Dans notre contexte, on privilégie la découverte d'un grand nombre de solutions crédibles, c'est-à-dire de formes SHACL. Notre objectif est donc de conserver un coefficient de diversité très haut. Enfin, étant donné $\mathcal{P}_i$ et $\mathcal{P}_{i+1}$ la population aux générations $i$ et $i+1$, le **taux de développement de la population** est égale au nombre d'individus différents entre les deux populations divisé par la taille totale de la population, soit $\frac{|\mathcal{P}_{i+1} \setminus \mathcal{P}_i|}{|\mathcal{P}_{i+1}|}$. Cet indicateur met en lumière la capacité de notre algorithme à explorer l'espace de solution et, par conséquent, à la découverte de solutions satisfaisantes. Un taux de développement modérément élevé et constant traduit une évolution progressive et saine d'une population.

## 4.2 Fonction de fitness basée sur la validation SHACL probabiliste

Pour que notre algorithme d'évolution grammaticale produise itérativement une population de formes SHACL exprimant certaines contraintes de domaine qui sont implicites dans un ensemble de données RDF, nous proposons une fonction de fitness basée sur une mesure d'acceptabilité d'une forme combinée avec le cadre probabiliste présenté dans la section 3.1.

La mesure d'**acceptabilité** d'une forme SHACL $s$, que l'on note $A(s)$, vis-à-vis d'un ensemble de données RDF, dépend du taux d'erreur observé lors de la validation de cet ensemble de données RDF par rapport à $s$, qui dépend lui-même directement de la proportion d'erreur physiologique théorique $p$. La valeur de $A(s) \in [0,1]$ est donnée par la formule suivante :

$$A(s) = \begin{cases} 1 & \text{si } \hat{p_s} \leq p \;\; ou \;\; X_s^2 \leq \chi^2_{k-1;\alpha} \quad \text{(voir les équations 2 et 4)}, \\ \\ \frac{L_{|\upsilon_s^-|}}{P(X=|\upsilon_s| \times p)} & \text{sinon} \quad \text{(voir l'équation 1)}. \end{cases} \tag{5}$$

Concernant le test d'hypothèse, nous considérons une marge d'erreur $\alpha = 0,05$. Par conséquent, la *valeur critique* est $\chi^2_{k-1;0,05} = 3,84$. Lorsque $A(s) = 1$, $s$ est acceptable et il est donc très probable que celle-ci soit sélectionnée parmi les meilleurs individus de la population. De ce fait, elle capture (probablement) certaines connaissances du domaine extraites de l'ensemble de données RDF. Dans le cas où l'hypothèse nulle est rejetée, $A(s) \neq 1$ et $s$ n'est pas acceptable, mais elle peut être prise en compte dans l'algorithme d'évolution grammaticale pour les opérations de croisement ou de mutation. À cette fin, $A(s)$ est égal à la vraisemblance de $s$ normalisée par la valeur maximale de la fonction de masse de probabilité pour une distribution binomiale $X \sim B(|\upsilon_s|, p)$. Cette normalisation permet de répartir davantage les valeurs $A(s)$ dans l'intervalle $[0,1]$ contrairement aux valeurs de vraisemblance, et ainsi de ne pas

pénaliser de manière trop importante des individus qui sont "proches" d'être acceptables mais dont la vraisemblance est faible (par exemple : $0,03$). Plus le taux d'erreur observé est faible par rapport au taux théorique, plus la valeur de $A(s)$ est élevée.

Le calcul du **score de fitness** d'un individu traduisant une forme SHACL $s$, que l'on note $F(s)$, vis-à-vis d'un ensemble de données RDF, combine son acceptabilité $A(s)$ et la cardinalité de ses confirmations $|\upsilon_s^+| : \forall s \in P, \quad F(s) = |\upsilon_s^+| \times A(s)$.

# 5 Expériences

## 5.1 Configuration

Le cadre expérimental se porte sur l'extraction de formes SHACL candidates à partir de l'ensemble de données *CovidOnTheWeb* [4] (Michel et al., 2020) produit à partir du jeu de données *COVID-19 Open Research Dataset (CORD-19)*. Il décrit un certain nombre d'articles scientifiques auxquels sont liés un certain nombre d'entités nommées alignées avec des entités *Wikidata*. Le tableau 1 présente les caractéristiques d'un sous-ensemble des données RDF considérées pour les expériences. Celui-ci contient 18,79% des articles et 0,01% des entités nommées de la base complète. Nous considérons l'extraction de formes SHACL représentant des règles d'association entre des entités nommées, comme celles issues des travaux de Cadorel et Tettamanzi (2020) extraites à partir d'un sous-ensemble de données de *CovidOnTheWeb* différent du nôtre. Cette expérimentation permet de mettre en évidence les capacités de l'approche à extraire des formes malgré l'incomplétude et les erreurs inhérentes de ce jeu de données Felin et al. (2023). De plus, le domaine médical permet d'obtenir des exemples de règles interprétables dont l'intérêt peut être facilement saisi. Enfin, la taille de la base de connaissances permet d'illustrer l'approche sur un exemple réel et non-trivial. Nous utilisons la grammaire BNF présentée dans la Figure 2 pour générer ces formes candidates. Celles-ci auront donc une structure phénotypique similaire à l'exemple présenté dans la figure 3. Nous avons utilisé le moteur de validation probabiliste SHACL implémenté dans le logiciel *Corese* [5]. Nous considérons la proportion d'erreur théorique $p = 0,5$ sur la base des résultats obtenus par Felin et al. (2023) sur le même jeu de données.

| # triplets RDF | # articles | # entités nommées | moyenne d'entités nommées par article |
|---|---|---|---|
| 226 647 | 20 912 | 6 331 | 10,52 |

TAB. 1 – *Caractéristiques du sous-ensemble de données RDF extraites depuis le graphe de données RDF CovidOnTheWeb.*

Les valeurs des paramètres des algorithmes évolutionnaires sont indiquées dans le tableau 2. Nous avons expérimenté 10 exécutions de notre algorithme par taille de population considérée, soit 30 exécutions au total. Le **code source** permettant l'exploration de formes SHACL candidates, le jeu de données RDF utilisé et l'ensemble des résultats sont disponibles dans un dépôt public [6].

4. https://github.com/Wimmics/CovidOnTheWeb
5. https://github.com/Wimmics/corese
6. https://github.com/RemiFELIN/RDFMining/tree/egc_2024

```
@prefix : <http://www.example.com/myDataGraph#> .
@prefix sh: <http://www.w3.org/ns/shacl#> .
@prefix entity: <http://www.wikidata.org/entity/> .

:1 a sh:NodeShape ;
    sh:targetClass  entity:Q424178 ;
    sh:property [
        sh:path rdf:type ;
        sh:hasValue entity:Q29548 ; ] .
```

FIG. 3 – *Exemple d'une forme SHACL traduisant une règle d'association où* `entity:Q424178` *("membrane raft") est l'antécédent de* `entity:Q29548` *("cell membrane") (conséquent).*

| **Paramètres** | **Valeur(s)** |
|---|---|
| $\|P\|$ | $\{200, 500, 1000\}$ |
| Effort (respectif) | $\{40000, 100000, 200000\}$ |
| # générations | 200 |
| Proportion de la sélection *élitiste* | 20% |
| Type de sélection *recombinaison* (proportion) | Sélection par tournois (40%) |
| Proportion d'individus composant les *tournois* | 25% |
| Type de crossover (proportion) | Croisement sur un point (75%) |
| Type de mutation (proportion) | Mutation par "retournement" d'un entier (5%) |
| Niveau de confiance $\alpha$ (TH) | 5% |
| Taux d'erreur $p$ (TH) | 50% |

TAB. 2 – *Paramètres globaux de l'algorithme (TH = Test d'Hypothèses, Retournement = génération aléatoire d'un nouvel entier).*

Les expériences ont été réalisées sur un serveur équipé d'un processeur Intel(R) Xeon(R) CPU E5-2637 v2 à une vitesse d'horloge de 3,50 GHz, avec 172 Go de RAM, 1 To d'espace disque fonctionnant sous le système d'exploitation Ubuntu 20.06.4 LTS 64 bits.

## 5.2 Résultats

**Formes candidates acceptées.** L'analyse des résultats montre que notre approche a permis la découverte d'un nombre significatif de formes SHACL candidates automatiquement acceptées : 2570 au total (voir le tableau 3). Plus la taille de la population est grande et plus nous obtenons de formes candidates acceptables, ce qui semble logique à nombre de générations égal. Ces formes sont majoritairement issues de la population élitiste mais une faible proportion est issue de la population destinée à évoluer, ce qui démontre un processus d'évolution efficace. Une majorité des formes sont acceptées à l'issue de tests d'hypothèses (59,1%), montrant l'impact majeur de ces tests dans le processus de décision.

La fouille de formes SHACL a abouti à la découverte de candidates *surprenants* dans le sens où elles impliquent des cardinalités relativement faibles et des violations parfois plus

| \|P\| | 200 | 500 | 1000 | Total |
|---|---|---|---|---|
| # formes acceptées | 301 | 847 | 1422 | 2570 |
| → faisant partie de l'élite | 269 (89,4%) | 738 (87,1%) | 1236 (86,9%) | 2243 (87,3%) |
| → avec TH | 166 (55,1%) | 520 (61,4%) | 833 (58,6%) | 1519 (59,1%) |
| RC | 22,25 | 18,32 | 16,57 | 19,05 |
| # violations | 7,56 | 7,67 | 7,67 | 7,63 |
| vraisemblance (%) | 15,73 | 14,54 | 13,5 | 14.59 |
| généralité (%) | 0,01 | 0,008 | 0,007 | 0,008 |
| Temps d'exécution (min) | 83,9 | 215 | 622,7 | 7280,3 |

TAB. 3 – *Résumé de l'ensemble des formes SHACL distinctes découvertes et acceptées (TH = Test d'Hypothèses ; RC = Référence de Cardinalité).*

nombreuses que le nombre théorique de violations (dont le taux est à 50%). Par exemple, nous pouvons citer la forme SHACL représentant la règle (`lung cancer` → `lung`) pour laquelle on observe 52% d'exceptions parmi les 50 triples testés. Cette forme a donc été accepté après un test d'hypothèse. On peut logiquement valider le fait qu'un *cancer du poumon* implique un lien direct avec *poumon* et attester de la validité de cette forme. Néanmoins, compte tenu du nombre important de formes SHACL acceptées et de la complexité de la quasi-majorité des règles sous-jacentes, nous suggérons une analyse approfondie réalisée par un expert afin d'apporter une validation finale pour chacune de ces formes.

On remarque également certaines règles pour lesquelles on observe une équivalence entre l'antécédent et le conséquent, comme par exemple (`cytokine` → `cytokine`), donnant ainsi des formes SHACL *triviales* et peu intéressantes. Ces phénotypes sont issus d'un ensemble de codons qui renvoient vers le même caractère phénotypique (c'est-à-dire la même classe). Toutefois, ce phénomène est minimal car observé chez seulement 96 individus parmi les 2570 individus découverts, soit 3,74% des cas. De plus, le filtrage de ce type d'individus peut impacter de manière négative l'évolution d'une population en réduisant l'exploration notamment avec des grammaires plus complexes. Nous recommandons donc de conserver la possibilité de découvrir ce type d'individus étant donné leur fréquence peu élevée.

**Performances.** Les expériences montrent que notre approche est capable de conserver un coefficient de diversité moyen à 100% au fur et à mesure que la population évolue (voir les figures 4a). Le taux de développement de la population est globalement très important au début de l'exploration (remplacement important des candidats *très mauvais*) jusqu'à converger vers une valeur fixe. Cette valeur semble légèrement augmenter avec la taille de la population, tandis que les fluctuations s'atténuent. Ce phénomène tend à expliquer l'évolution du score de fitness moyen d'une population (voir les figures 4b) où l'on s'aperçoit que les sauts au niveau des valeurs sont de moins en moins fréquents sans toutefois impacter le score moyen final.

Le temps d'évaluation moyen des formes SHACL candidates est très faible : le temps CPU moyen dédié à l'évaluation SHACL probabiliste pour un candidat est de 13,16 ms. La figure 5 montre une croissance du temps d'exécution avec le nombre de violations observées lors de l'évaluation SHACL probabiliste. On note un temps CPU maximum de (seulement) 401 ms. atteint pour une forme candidate largement rejetée par le test d'hypothèse (1098 violations parmi les 1123 triplets testés).

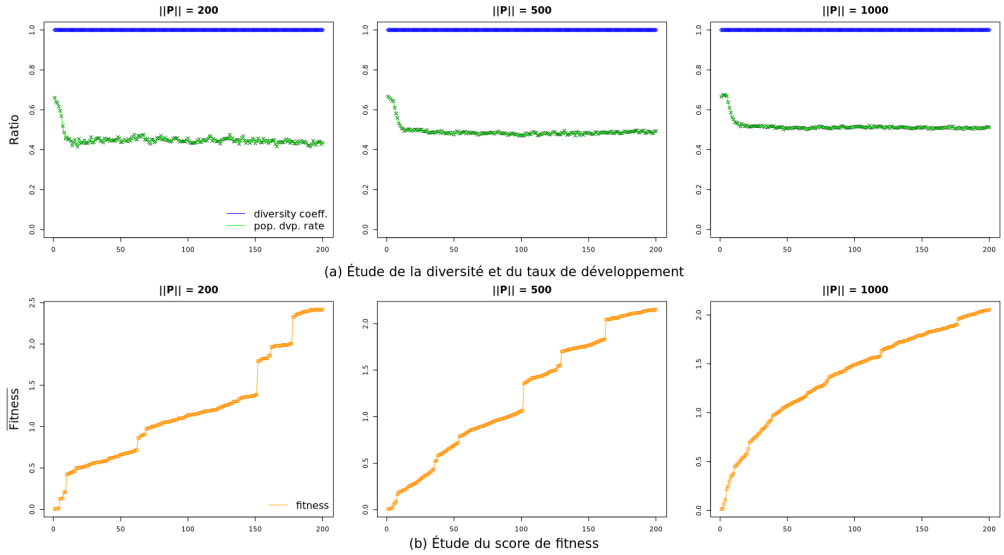


FIG. 4 – *Statistiques moyennes des résultats obtenus en fonction des générations, à partir de 10 exécutions, par tranche de population.*

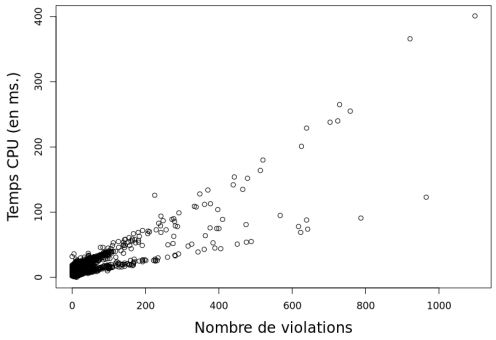


FIG. 5 – *Temps d'exécution CPU des formes SHACL découvertes en fonction du nombre de violations observées.*

**Couverture.** Nous avons également estimé la couverture de nos expériences, c'est-à-dire le nombre de formes acceptables trouvées par rapport au nombre de formes acceptables potentiellement trouvables. Pour cela, notons $\Omega$ l'ensemble des paires (*antécédent*, *conséquent*) d'entités nommées distinctes extraites de notre sous-ensemble de données RDF. Sa taille est $|\Omega| = 6\ 331 \times 6\ 330 = 39\ 627\ 000$. Pour estimer le nombre de paires acceptables dans cet ensemble dans les conditions expérimentales présentées ci-dessus, nous utilisons un *échantillon aléatoire* représentatif de l'espace des solutions $\Omega' \subseteq \Omega$. Afin de déterminer la cardinalité minimale de $\Omega'$ pour assurer la représentativité, on pose le niveau de confiance selon

la distribution normale standard $z$, l'ignorance quant à la probabilité $p'$ qu'un candidat étudié soit acceptable (c'est-à-dire $p' = 0,5$), un niveau de confiance à $99\%$ (ce qui implique $z \approx 2,58$) et une marge d'erreur tolérée $m$ de $2\%$, nous utilisons la formule de Cochran : $|\mathbf{\Omega'}| = \frac{z^2 \times p \times (1-p)}{m^2} = \frac{2{,}58^2 * 0{,}5^2}{0{,}02^2} \approx \mathbf{4\,161}$.

Nous avons généré un ensemble $\Omega'$ de 4 161 formes SHACL distinctes de manière aléatoire que nous avons ensuite évaluées sur le sous-ensemble de données RDF. Seules 2 formes dans $\Omega'$ sont acceptées, soit 0,05% du total. Cela nous permet d'estimer le nombre de formes acceptables dans $\Omega$ à $||\Omega|| \times 0,0005$. Lors de nos expériences, nous avons pu découvrir 2 570 formes SHACL distinctes acceptables. Le **rappel** $R$ caractérisant la couverture de cet ensemble de solutions par rapport à l'ensemble de solutions global est donc : $\mathbf{R} = \frac{2\,570 \times 100}{|\Omega| \times 0{,}0005} \approx \mathbf{12,97}\%$.

Ce résultat est donc satisfaisant eu égard à l'effort investi et la difficulté de la tâche. De plus, un nombre d'exécutions plus important conduirait à une exploration plus ample de l'espace de solutions $\Omega$ et accroîtrait les chances de découvrir de nouvelles formes acceptables.

# 6 Conclusion

Dans cet article, nous avons proposé un cadre utilisant une méthode d'évolution grammaticale pour extraire des formes SHACL candidates d'un jeu de données RDF à partir d'une grammaire BNF des formes définie manuellement. Ce cadre utilise un processus probabiliste de validation SHACL avec une mesure d'acceptabilité et une fonction de fitness pour évaluer les formes candidates et retenir les meilleures. Les expériences montrent que notre approche capture des formes SHACL intéressantes décrivant des contraintes de domaine. En outre, elle fournit un moyen efficace de mettre en évidence des structures subtiles dans les données RDF, de pondérer les erreurs que celles-ci peuvent impliquer et adapter la fouille de ces formes en conséquence. Nos futures recherches se concentreront sur l'étude de la mise à l'échelle sur des ensembles de données plus importants tels que DBpedia, des grammaires plus expressives permettant la découverte de formes plus complexes, et une évaluation qualitative approfondie des formes par des experts.

**Remerciements.** Ce travail a été financé partiellement par le projet "Investissements d'Avenir" 3IA Côte d'Azur financé par l'Agence Nationale de la Recherche (ANR-19-P3IA-0002).

# Références

Baeck, T., D. Fogel, et Z. E. Michalewicz (2000). *Evolutionary Computation 1 : Basic Algorithms and Operators (1st ed.)*. CRC Press.

Boneva, I. et al. (2019). Shape designer for shex and SHACL constraints. In *ISWC (Satellites)*, Volume 2456 of *CEUR Workshop Proceedings*, pp. 269–272. CEUR-WS.org.

Cadorel, L. et A. G. B. Tettamanzi (2020). Mining RDF data of COVID-19 scientific literature for interesting association rules. In *WI/IAT*, pp. 145–152. IEEE.

Cimmino, A. et al. (2020). Astrea : Automatic generation of SHACL shapes from ontologies. In *ESWC*, Volume 12123 of *Lecture Notes in Computer Science*, pp. 497–513. Springer.

Felin, R. et al. (2023). A Framework to Include and Exploit Probabilistic Information in SHACL Validation Reports. In *ESWC*.

Fernández-Álvarez, D. et al. (2018). Inference of latent shape expressions associated to dbpedia ontology. In *ISWC (P&D/Industry/BlueSky)*.

Fernández-Álvarez, D. et al. (2022). Automatic extraction of shapes using shexer. *Knowl. Based Syst. 238*, 107975.

Keely, A. (2020). shaclgen.

Kontokostas, D. et H. Knublauch (2017). Shapes constraint language (SHACL). W3C recommendation, W3C.

Michel, F. et al. (2020). Covid-on-the-web : Knowledge graph and services to advance COVID-19 research. In *ISWC (2)*, pp. 294–310. Springer.

Mihindukulasooriya, N. et al. (2018). RDF shape induction using knowledge base profiling. In *SAC*, pp. 1952–1959. ACM.

Nguyen, T. H. et A. G. B. Tettamanzi (2019). An evolutionary approach to class disjointness axiom discovery. In *WI*, pp. 68–75. ACM.

Omran, P. et al. (2022). Learning shacl shapes from knowledge graphs. *Semantic Web 14*, 1–21.

O'Neill, M. et C. Ryan (2001). Grammatical evolution. *IEEE Trans. Evol. Comput. 5*(4), 349–358.

Pandit, H. et al. (2018). Using ontology design patterns to define shacl shapes. In *WOP@ISWC*, Monterey California, USA, pp. 67–71.

Rabbani, K. et al. (2022). SHACL and shex in the wild : A community survey on validating shapes generation and adoption. In *WWW (Companion Volume)*, pp. 260–263. ACM.

Saleh, D. R. et al. (2022). On generating SHACL shapes from collective collection of plant trait data. In *IC3INA*, pp. 326–330. ACM.

## Summary

SHACL is the W3C recommendation to represent constraints on RDF graphs called *shapes*, used to validate RDF data. Creating shapes capturing domain constraints is a very tedious task. That is why recent work addresses the problem of automatic shapes generation. In this context, we propose an approach combining grammatical evolution and a probabilistic framework to generate candidate SHACL shapes and validate RDF graphs against candidate shapes. The approach is validated by mining SHACL shapes representing association rules from an RDF dataset. The results show the potential and generality of the proposed approach.