Making constraint solvers more usable : overconstraint problem

Etude d'article

Olivier THOMAS

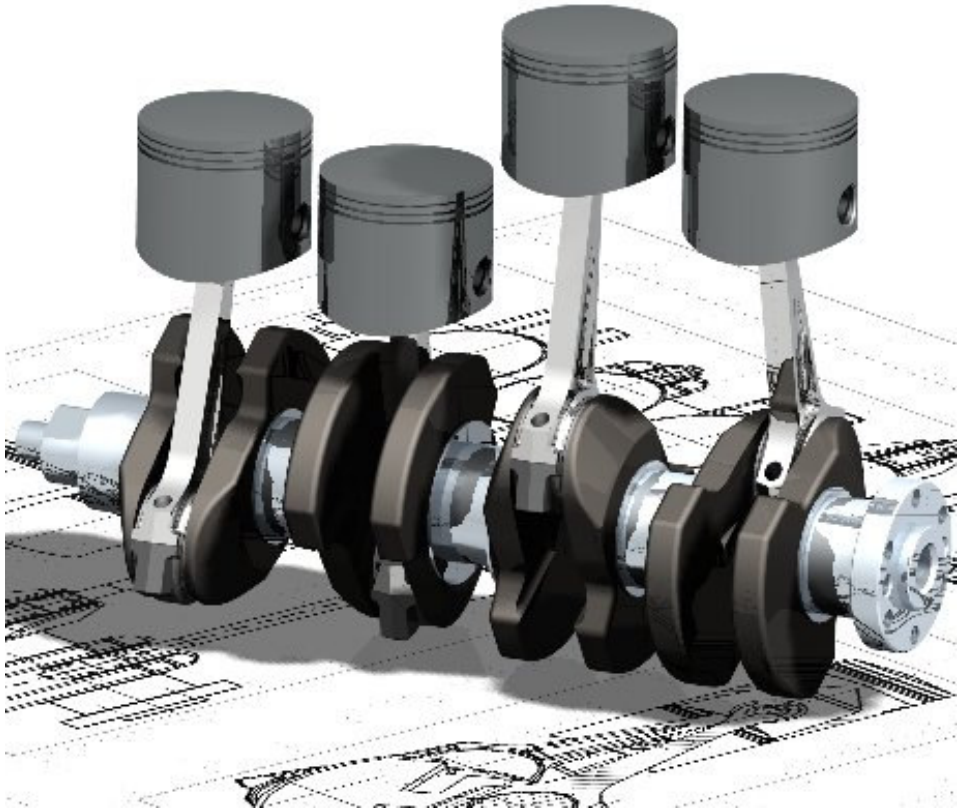
Plan

Problème des systèmes géométriques sur-contraints

- Introduction / Contexte
- 1. Généralités sur les solveurs
- 2. Une première solution
- 3. Une meilleure solution
- Conclusion

Introduction

- CAD : Computer-Aided Design
- Plan avec cotes → objet « concret »
- Ingénierie
  - Automobile
  - Aéronautique
  - …
- Enseignement
  - Cours de géométrie au lycée
  - Préparation au CAPES/Aggregation
  - …
- Design
  - Architecture
  - …
- Croquis coté = système géométrique contraint



Introduction

- Système géométrique contraint =
  ensemble fini de primitives géométriques
  + ensemble fini de relations entre ces objets
  - Primitives :
    - Dans le plan : point, ligne, conique.
    - Dans l'espace : point, ligne, plan, cylindre, sphère.
  - Relations : contraintes
    - logiques : incidence, tangence, perpendicularité, …
    - métriques : distance, angle, rayon, …
- Objet « concret » = solution au système contraint
  - i.e. une classe d'instanciations valides des primitives géométriques satisfaisant toutes les contraintes.
- → Systèmes de résolution de contraintes : solveur

Contexte

Il existe de nombreux systèmes de résolution de contraintes :

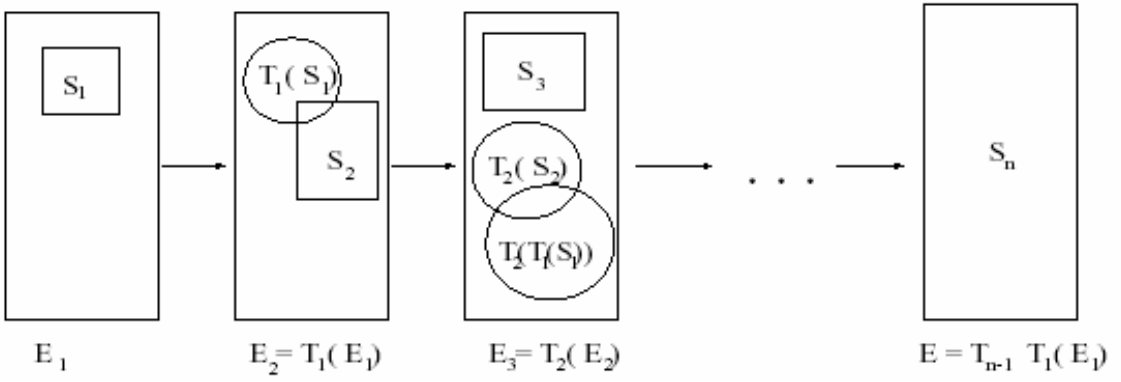
- Les débuts :
  - SketchPad : Ivan Sutherland, MIT, 1963
  - Geometry Machine, Gelernter, 1963
  - DAC-1 (Design Augmented by Computer), 1964
  - UNISURF, Bézier, Renault, 1971
  - …
- Aujourd'hui
  - Solveur algébrique de Wu : pseudo-division euclidienne
  - PTC : système hybride : objets BREP ←→ CSG
  - AutoCAD : notion de blocs = objets réutilisables
  - CATIA : Dassault, système hybride + plateforme de recherche en interne. Utilisé par Boeing pour le 777!
  - …
- Pas de méthodologie générale et unifiée : solutions particulières adaptées à un problème

1. Généralités
Les solveurs

- Système géométrique :
  - Bien contraint : on peut le résoudre
  - Surcontraint, souscontraint : revenir à un système bien contraint
- Un « bon » solveur :
  - Général : bon niveau d'abstraction
  - Efficace : complexité linéaire en o(taille(système contraint))
  - Résolution des ambiguïtés/incohérences
    - Un système qui « a l'air » bien contraint ne l'est en fait pas
  - Désambiguïsation en cas de problème sur-contraint :
    - Sans expertise mathématique de l'utilisateur!
    - Générale : pas de limitations arbitraires
    - Reproductible : comportement déterministe
    - Mise à jour dynamique du système contraint

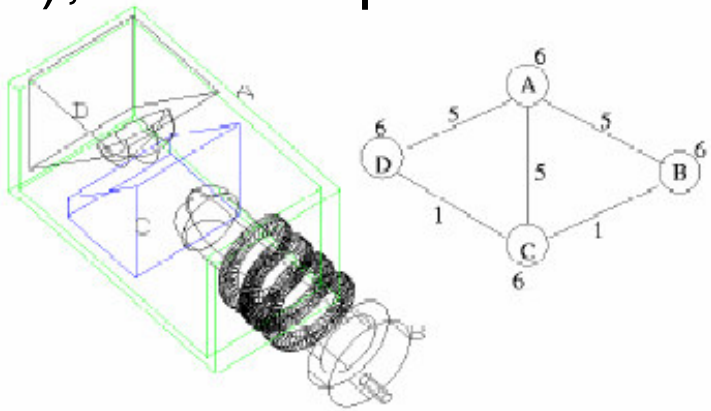
1. Généralités
Les plans DR

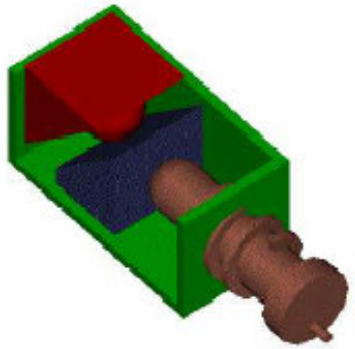
- Efficacité de la résolution du problème
  - Naïvement : problème combinatoire → complexité exponentielle
  - Découper le problème pour réduire la complexité
    - **Plan de Décomposition-Recombinaison (plan DR)**
  - Plan DR optimal : minimise la taille du plus grand sous-système
- Méthode générale : récursive
  - Trouver un petit sous-système bien contraint S_i (*cluster*) du système E_i ($E_1 = E$)
  - Résoudre S_i (méthode algébrique ou analytique)
  - Substituer S_i par $T_i(S_i)$ (T_i : simplifieur) dans E_i
    - ie simplifier E_i : $T_i(E_i) = E_{i+1}$



1. Généralités
Graphes de contraintes

- S'abstraire de la géométrie, être général
- Graphe de contraintes G = (V, E, w)
  - V = sommets : objet géométriques
    - w(v), v ∈ W = poids de v : #DoF de l'objet v
  - E = arrêtes : contraintes
    - w(e), e ∈ E = poids de e : #DoF enlevées par e





1. Généralités
Graphes de contraintes : densité

- Sous graphe A de G dense si :

$$\sum_{e \in A} w(e) + D \geq \sum_{v \in A} w(v)$$

  - D = DoF associés au sous graphe dense
    - Plan euclidien : D = 3, i.e. à une translation près
    - Espace euclidien : D = 6, .i.e. à une translation/rotation près
- → Densité d'un graphe A :

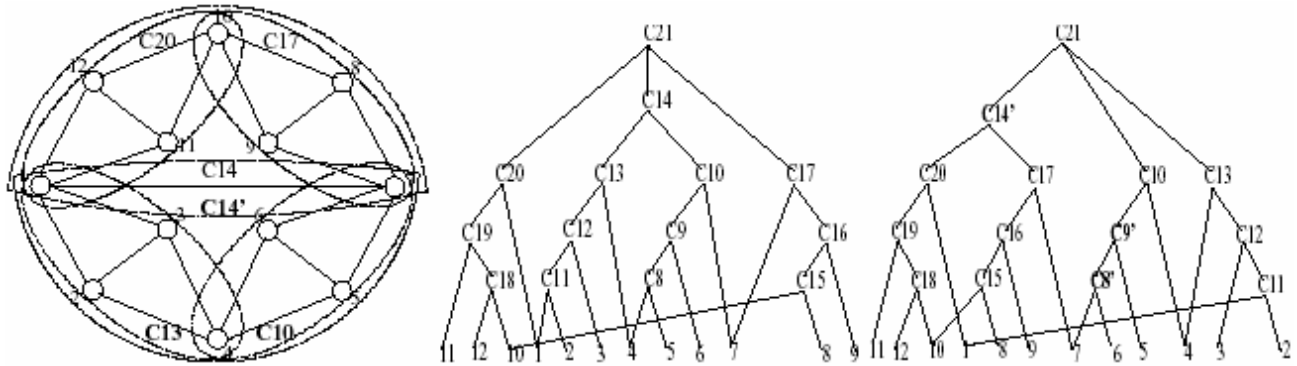
$$d(A) = \sum_{e \in A} w(e) - \sum_{v \in A} w(v)$$

1. Généralités
Graphes de contraintes : densité

- Soit G un graphe dense :
  - G surcontraint : d(G) > -D :
  - G bien contraint :
    - G dense
    - G a au moins un sousgraphe surcontraint
    - Reste dense si on remplace tous les sous graphes surcontraints par des graphes bien contraints
  - G bien surcontraint
    - si tous les sous graphes de G ont une densité ≤ -D
  - G souscontraint sinon.
- Graphe dense minimal si ne possède aucun sous graphe dense non trivial
  - **Tout sous graphe dense minimal est bien (sur)contraint !**

1. Généralités
Méthodes de désambiguïsation

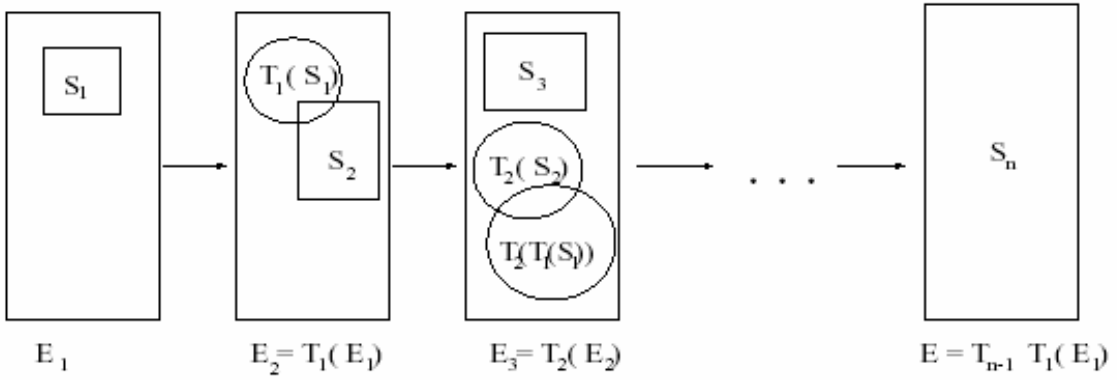
- Problème surcontraint → ambiguïté sur la (les) solution(s)
- Utiliser efficacement les plan DR pour guider la désambiguïsation
  - **Simplifie** le travail de l'utilisateur : solutions « locales » vs « globales »
  - → Besoin de m-à-j **facile**/**rapide** du plan DR si système évolue
  - → Efficacité de l'utilisation des plans DR : **problème ouvert**
  - → Aujourd'hui : utilisation d'**heuristiques** :
    - Conserver l'orientation relative des éléments géométriques du croquis de départ
    - Nouveau croquis : souscontraint, représentation visuelle des contraintes à modifier
    - Inférences heuristiques « futées » de contraintes supplémentaires à partir du croquis de départ et d'informations données par l'utilisateur



2. Une première solution
Méthode

- Approche mixte ascendente / descendante
  - Méthode de rigidification progressive → notion de cluster : ensemble rigide
  - ↑ : propagation de contraintes à partir d'un cluster minimal
  - ↓ : simplification du problème par un cluster

1. Trouver et isoler un sous graphe dense minimal
2. Simplifier le problème en le « divisant » par le sous graphe dense minimal (*cluster*)
3. Recommencer



2. Une première solution
2.1 Trouver / isoler :

- Sous graphe dense
  - Init : sous graphe vide G' de G
  - Rajouter un par un des sommets v de G dans G'
  - Pour chaque sommet v rajouté, « distribuer v » en considérant toutes les arrêtes incidentes e_i, i.e.
    - « Problème du mariage parfait » :
      graphe de flot G* associé à G, méthode des chaînes améliorantes :
      $\forall$ e_i incidentes à v, distribuer le flot $w(e_i) + D + 1$ d'un bout à l'autre du graphe de flot
    - Si on peut distribuer toutes les arrêtes, G' n'est pas dense
    - Sinon G' est dense
- Sous graphe dense minimal
  - Prendre G' et lui enlever un sommet : obtention de G''
  - Ré appliquer méthode précédente : G'' est-il toujours dense?
    - Si oui : recommencer
    - Si non : G' sous graphe dense minimal : G' est un cluster!

2. Une première solution
2.2 Simplifier

- « Diviser » G_i par le cluster S_i pour simplifier G_i en G_{i+1}
- Notion de sommet « frontière »
  - Un cluster S_i interagit avec le reste du graphe par ses **sommets frontières**, les sommets adjacents à des sommets $\notin$ cluster S_i.
  - Les autre sommets de S_i s'appellent des **sommets noyaux**.
- Réduire les sommets noyaux de S_i à un unique sommet (simplification, condensation) :
  - Connecter les sommets frontières au nouveau sommet par des arrêtes, dont le poids est la somme des « anciennes » arrêtes
  - Poids du sommet noyau choisi tel que la densité de S_i simplifié soit égale à –D
- Répéter jusqu'à obtention d'un S_m qui corresponde au graphe G_m restant
- **On vient de simplifier le problème par des sous problèmes bien contraints jusqu'à ce que le problème soit bien contraint et solvable!**

2. Une première solution Discussion

- Ne répond pas aux critères d'un bon solveur :
  - Complexité trop élevée : $o(|V|^2 * (|V| + |E|))$
    - Sous tend une « mauvaise » utilisation du plan DR
  - Pas de calcul incrémental des contraintes en conflit (arrêtes réductibles)
    - Une arrête d'un sous graphe G' de G est **réductible** si on peut décrémenter son poids de 1 sans rendre G' souscontraint. De plus G est dit **1-surcontraint**.
  - Convivialité : ne minimise pas le travail de l'utilisateur
    - On devrait pouvoir obtenir les contraintes à modifier (arrêtes réductibles) en fonction d'un sous système donné
    - De façon optimale : ne parcourir que les clusters pertinents
  - Pas de m-à-j efficace après avoir isolé une arrête réductible :
    - N'inspecter que les clusters affectés par la réduction
- → Proposer une meilleure solution : améliorer la complexité et la convivialité

3. Une meilleure solution
Méthode

- Traverser le plan DR de haut en bas pour calculer incrémentallement le sous ensemble minimal de contraintes en conflit
  - Stocker les résultats intermédiaires
  - → Pas besoin de recalculer tout le plan à chaque modification
- Sélectionner les contraintes du cluster choisi par l'utilisateur
  - Critère de convivialité : approche naturelle, limite le nombre de contraintes, énumération des solutions de façons graphique
- Isoler les données concernant le plan DR qui peuvent être stockées/maintenues
- M-à-j automatique et efficace du plan DR
  - Réorganisation minimale en terme d'opérations une fois les contraintes en conflit résolues
  - Utilise les sommets frontière

3. Une meilleure méthode Extensions / Modifications

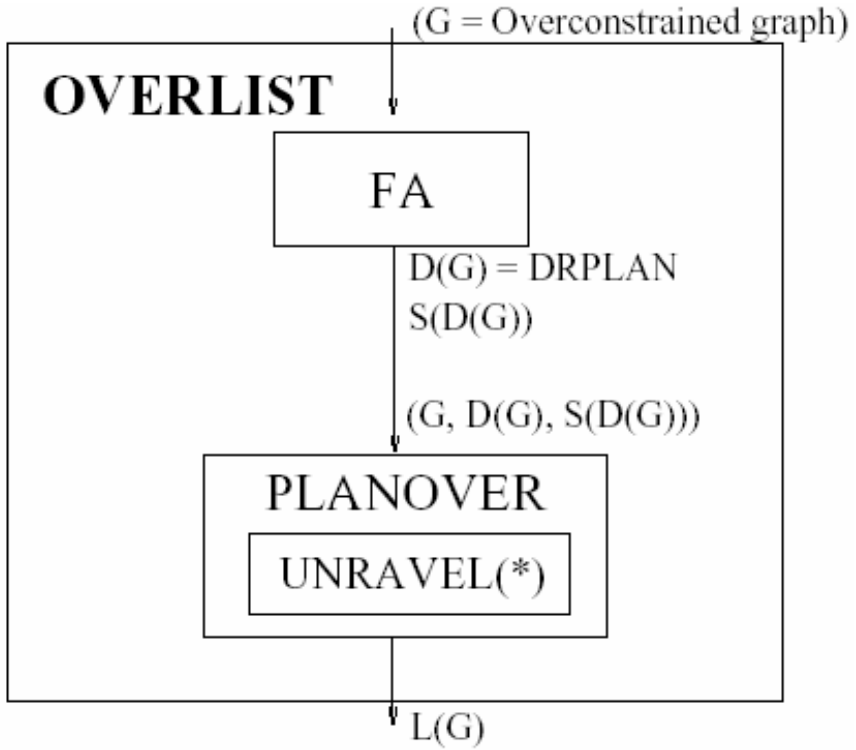
- Permet (modulo des modifications minimes) de répondre à des questions de l'utilisateur
  - « Cette arrête est-elle réductible? » (i.e. est-elle en conflit)
  - « A partir d'un cluster C, donne moi la liste des arrêtes réductibles de C »
  - Rend le solveur convivial
    - Point TRES important de la méthode!

3. Une meilleure méthode Extensions / Modifications

- Passer aux problèmes de graphes k-surcontraints
  - Méthode k=1 efficace lors d'une construction incrémentale (*from scratch*)
  - Mais si on mélange un ensemble d'objets prédéfinis?
    - Classique en entreprise : on réutilise des constructions déjà existantes (pièces disponibles, par ex : pistons, cylindres) pour créer une plus grosse construction (par ex : nouveau moteur)
    - → Graphes k-surcontraints
      - Méthode pour k=1 **raisonnablement** applicables de façon incrémentale sur les graphes k-surcontraints

3. Une meilleure solution
Aperçu du pseudo-code

- OVERLIST(G)
  - G = graphe de contraintes
  - Génère
    - Liste des arrêtes réductibles de G
- FA(G) : Frontier Analysis
  - Génère :
    - D(G) : un plan DR à partir de G
    - S(D(G)) : le plus petit graphe 1-bien-surcontraint non trivial de D(G)
- PLANOVER(G, D(G), S(D(G))
  - Appelle les 2 variantes UNRAVEL
  - « Aiguillage »
- UNRAVEL / UNRAVEL*
  - A partir d'un sous plan DR D'(G) de D, D'(G) enraciné en S(D(G))
  - Génèrent **incrémentalement** une liste des arrêtes réductibles de G



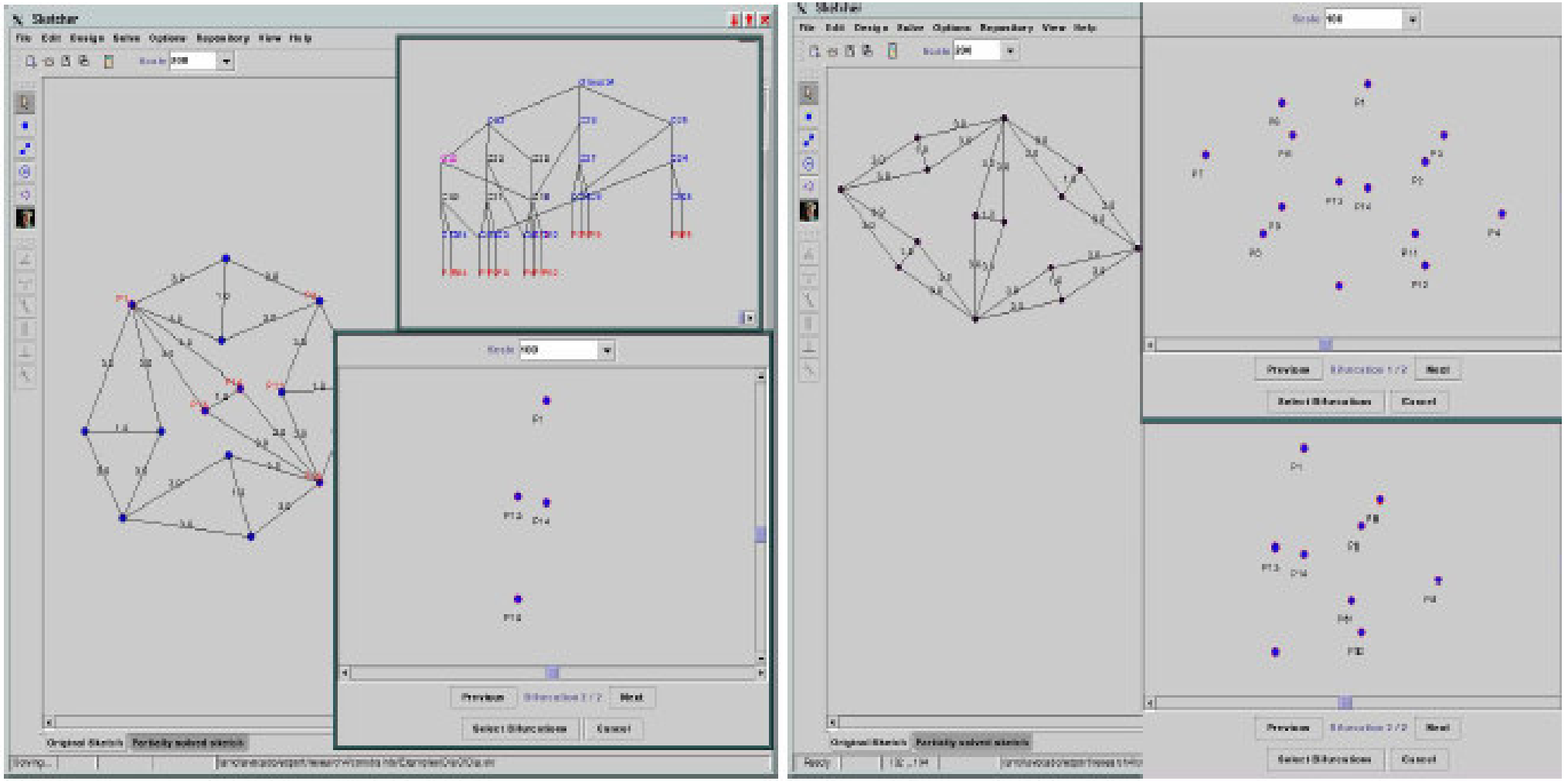
Principal apport de l'article

3. Une meilleure méthode Discussion

- Générale et flexible
  - À condition d'évacuer les cas triviaux
- Efficace
  - Complexité bornée par
    o(min { l_c*(|V| + |E|), max { $|V|^2$, |E| } })
    - Avec l_c =longueur de l'exploration du plan DR
  - En pratique : **complexité linéaire**, bornée par o(|E|)
  - Utiliser des structures de données TRES optimisées (pas trivial : sujet d'une parution dédiée!)
- Convivialité de la méthode
- Répond aux critères d'un bon solveur… (définis par les auteurs ;))

- … **mais** problème du plan DR optimal évacué
  - Problème ouvert
  - Nécessité de fournir un bon plan DR à l'algorithme

3. Une meilleure solution
Implantation : FRONTIER

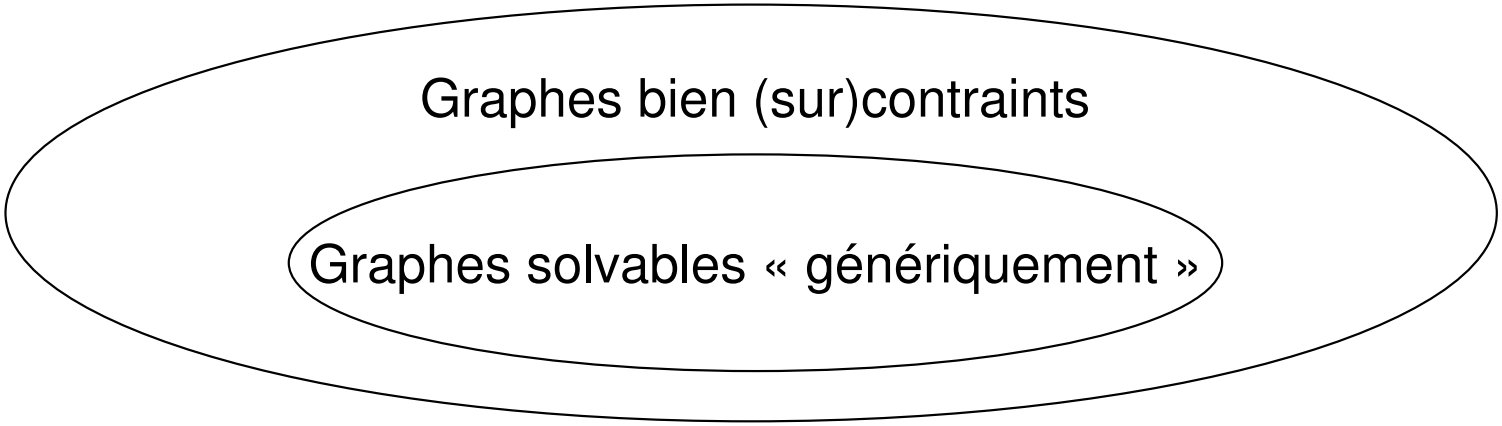


Conclusion

- Qu'apporte le travail des auteurs :
  - Identifier **clairement** et **efficacement** le sous ensemble minimal de contraintes en conflit
    - À partir d'un plan DR donné !
  - Désambiguïsation « graphique » efficace et **résolution** des incohérences/ambiguïtés
    - Les systèmes actuels se contentent de les détecter et demandent à l'utilisateur de rajouter/enlever des contraintes sans le guider (expertise en systèmes contraints)
  - Mise en application : modeleur CAD FRONTIER
    - GNU : plateforme de recherche « accessible » (comparé à CATIA par exemple…)
  - Limiter l'utilisation d'heuristiques, mieux formaliser le problème
    - Aujourd'hui, chaque développeur « bricole » ses solutions dans son coin : → essayer de généraliser
    - Formalisation → preuves de correction / terminaison

Mais …

- Ne fonctionne (très) bien que sur les graphes 1-surcontraint
  - Processus de design incrémental pas très intéressant pour une entreprise
  - Adaptation incrémentale à des graphes k-surcontraints
    → Attention de ne pas « noyer » l'utilisateur, à nouveau problème de convivialité
- Ne règle pas son compte au problème de la génération du plan DR optimal (solution(s) interne(s) testée(s) dans FRONTIER)
- Évacuation des cas « pathologiques » :
  - Cas triviaux traités comme des exceptions pour des raisons d'efficacité
  - Un peu ballot pour une méthode générale. (Efficacité ←→ Généralité)



Références

- Publication(s)
  - Christoph M. Hoffmann, Meera Sitharam, Bo Yuan, *Making constraint solvers more usable : overconstraint problem*, Avril 2003
- Cours de M. Dufourd et M. Schreck
  - sur les systèmes contraints et leurs résolution
- Sites WEB
  - FRONTIER : http://www.cise.ufl.edu/~sitharam/GNU/
  - AutoCAD : http://en.wikipedia.org/wiki/Autocad
  - CATIA : http://en.wikipedia.org/wiki/CATIA