Université de Montréal

Le réalisateur de texte jsRealB

Guy Lapalme

DIRO

rali

Génération de texte (Reiter-Dale 2000)

Pipeline pour le *Data to Text*

	Contenu	**Structure**
Macroplanning	1 Détermination du contenu	2 Organisation du contenu
Microplanning	4 Lexicalisation 5 Expression référentielle	3 Agrégation
Réalisation	6 Réalisation linguistique	7 Présentation

DIRO

rali

Génération de texte (Reiter-Dale 2000)

Pipeline pour le *Data to Text*

	Contenu	**Structure**
Macroplanning	1 Détermination du contenu	2 Organisation du contenu
Microplanning	4 Lexicalisation 5 Expression référentielle	3 Agrégation
Réalisation	6 Réalisation linguistique	7 Présentation

DIRO

rali

Approches à la génération

- À base de théories linguistiques
  - Systemic Functional Linguistic: KPML, Surge
  - Sens-Texte: RealPro, Forge, GenDR
- Apprentissage de bout en bout
  - GPT-*n*, T5
- Réalisateurs
  - Simple-NLG : Java
  - RosaeNLG : JavaScript / PUG Templates
  - CoreNLG : Python
  - ARRIA : templating engine

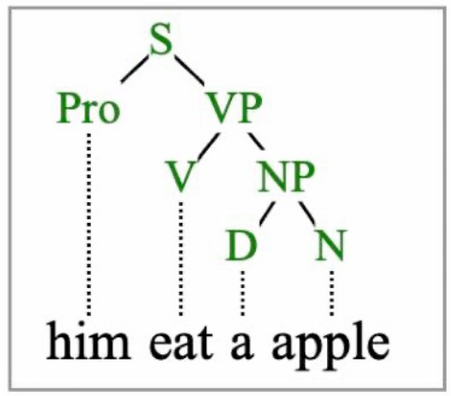
Que fait un réalisateur de texte ?

- Traite de multiples *détails*, mais *importants*
  - morphologie
  - accords
  - ordre des mots
  - élision
- jsRealB traite aussi
  - génération du HTML
  - variantes de phrases
  - anglais, français et même bilingues

Format de l'entrée

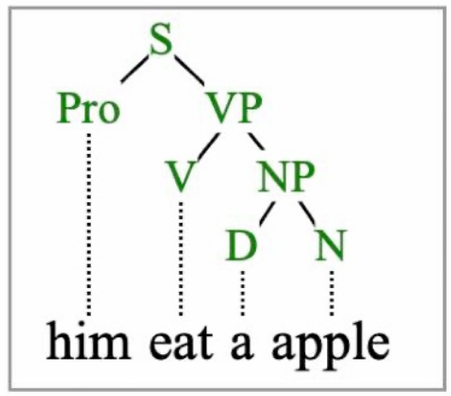
- Structures de constituants
  - *Terminaux* : `N(..), A(..), V(..),D(..),…`
  - *Syntagmes* : `NP(..), AP(..), VP(..), S(..),…`
  - *Options* : `.pe(..), .n(..), .t(..)`

```
S(Pro("him").c("nom"),
  VP(V("eat"),
     NP(D("a"),
        N("apple").n("p")).tag("em")))
```



Format de l'entrée

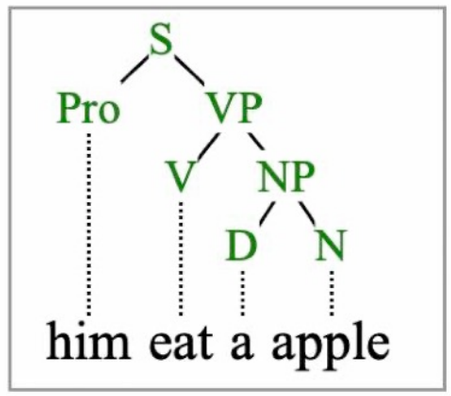
- Structures de constituants
  - *Terminaux* : `N(..), A(..), V(..),D(..),…`
  - *Syntagmes* : `NP(..), AP(..), VP(..), S(..),…`
  - *Options* : `.pe(..), .n(..), .t(..)`




```
S(Pro("him").c("nom"),
  VP(V("eat"),
     NP(D("a"),
        N("apple").n("p")).tag("em"))).toString()
```

Format de l'entrée

- Structures de constituants
  - *Terminaux* : `N(..), A(..), V(..),D(..),…`
  - *Syntagmes* : `NP(..), AP(..), VP(..), S(..),…`
  - *Options* : `.pe(..), .n(..), .t(..)`

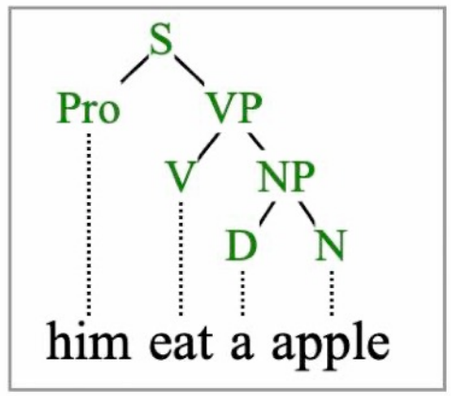



```
S(Pro("him").c("nom"),
  VP(V("eat"),
    NP(D("a"),
      N("apple").n("p")).tag("em"))).toString()
```

He eats apples.

Format de l'entrée

- Structures de constituants
  - *Terminaux* : `N(..), A(..), V(..),D(..),…`
  - *Syntagmes* : `NP(..), AP(..), VP(..), S(..),…`
  - *Options* : `.pe(..), .n(..), .t(..)`



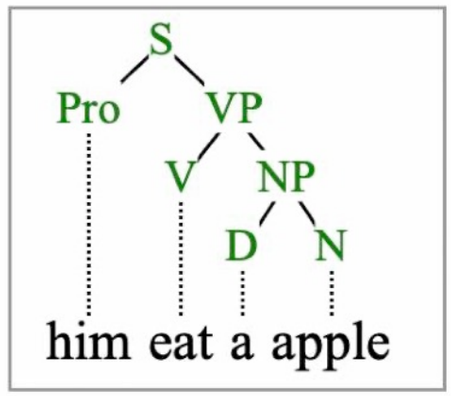

```
S(Pro("him").c("nom"),
  VP(V("eat"),
     NP(D("a"),
        N("apple").n("p")).tag("em"))).toString()
```

He eats apples.

```
{"phrase":"S",
 "elements":[{"terminal":"Pro","lemma":"him","props":{"c":"nom"}},
             {"phrase":"VP",
              "elements":[{"terminal":"V","lemma":"eat"},
                          {"phrase":"NP",
                           "elements":[{"terminal":"D","lemma":"a"},
                                       {"terminal":"N","lemma":"apple",
                                        "props":{"n":"p"}}],
                           "props":{"tag":[["em",{}]]}}]}],
 "lang":"en"}
```

Format de l'entrée

- Structures de constituants
  - *Terminaux* : `N(..), A(..), V(..),D(..),…`
  - *Syntagmes* : `NP(..), AP(..), VP(..), S(..),…`
  - *Options* : `.pe(..), .n(..), .t(..)`




```
S(Pro("him").c("nom"),
  VP(V("eat"),
     NP(D("a"),
        N("apple").n("p")).tag("em"))).toString()
```

He eats apples.

```
fromJSON({"phrase":"S",
          "elements":[{"terminal":"Pro","lemma":"him","props":{"c":"nom"}},
                      {"phrase":"VP",
                       "elements":[{"terminal":"V","lemma":"eat"},
                                   {"phrase":"NP",
                                    "elements":[{"terminal":"D","lemma":"a"},
                                                {"terminal":"N","lemma":"apple",
                                                 "props":{"n":"p"}}],
                                    "props":{"tag":[["em",{}]]}}]}],
          "lang":"en"}.toString()
```

SimpleNLG

```
Lexicon lexicon = new XMLLexicon();      // default simplenlg lexicon
NLGFactory nlgFactory = new NLGFactory(lexicon);

NPPhraseSpec np= nlgFactory.createNounPhrase("a", "apple");// create NP
np.setFeature(Feature.NUMBER,NumberAgreement.PLURAL);   // set plural
SPhraseSpec s = nlgFactory.createClause("he","eat", np);// create sentence
DocumentElement sentence = nlgFactory.createSentence(s);

Realiser realiser = new Realiser(lexicon);
NLGElement realised = realiser.realise(sentence);
System.out.println(realised.getRealisation());
```

```
from CoreNLG.DocumentConstructors import TextClass
class Content(TextClass):
    def __init__(self, section):
        super().__init__(section)
        self.text = self.free_text(
            "he",
            "eats",
            self.nlg_tags("em",text="apples")
        )
```

CoreNLG

DIRO

rali

Couverture lexicale

- Français
  - lexique (52 K entrées)
  - règles de déclinaisons et conjugaison
    - temps simples et composés, verbes réfléchis
- Anglais
  - lexique (34K entrées)
  - règles de déclinaisons et conjugaison
    - temps simples, progressif, perfect
    - comparatif et superlatif pour adverbes et adjectifs
- Ajout possible de nouvelles entrées de lexique

Couverture syntaxique

- accord entre déterminant, adjectif, nom
- accord entre sujet (`NP` ou `Pro`) et verbe
- pronominalisation

```
S(Pro("lui").c("nom"),
  VP(V("donner").t("pc"),
     NP(D("un"), N("pomme")).pro())
)
```

Couverture syntaxique

- accord entre déterminant, adjectif, nom
- accord entre sujet (`NP` ou `Pro`) et verbe
- pronominalisation

```
S(Pro("lui").c("nom"),
  VP(V("donner").t("pc"),
     NP(D("un"), N("pomme")).pro())
)
```

Il l'a donnée.

Variantes d'une structure

- Français (1248 possibilités)
  - négation, passif, progressif, réflexif, exclamatif
  - interrogatif (oui/non, (à)qui, (à)quoi, où, pourquoi, quand, comment, combien)
  - modalité (possibilité, permission, nécessité, obligation, volonté)
- Anglais (4992 possibilités)
  - negation, passive, perfect, progressive, réflexive, exclamative, contraction
  - interrogatif (yes/no, (to)who, (to)what, where, why, when, how, how much)
  - modality (possibility, permission, necessity, obligation, willingness)

```
S(Pro("lui").c("nom"),
  VP(V("donner").t("pc"),
     NP(D("un"), N("pomme")).pro())
).typ({"neg":true,"pas":true})
```

Variantes d'une structure

- Français (1248 possibilités)
  - négation, passif, progressif, réflexif, exclamatif
  - interrogatif (oui/non, (à)qui, (à)quoi, où, pourquoi, quand, comment, combien)
  - modalité (possibilité, permission, nécessité, obligation, volonté)
- Anglais (4992 possibilités)
  - negation, passive, perfect, progressive, réflexive, exclamative, contraction
  - interrogatif (yes/no, (to)who, (to)what, where, why, when, how, how much)
  - modality (possibility, permission, necessity, obligation, willingness)

```
S(Pro("lui").c("nom"),
  VP(V("donner").t("pc"),
     NP(D("un"), N("pomme")).pro())
).typ({"neg":true,"pas":true})
```

Elle n'a pas été donnée par lui.

Coordination

```
S(CP(C("et"),NP(D("le"),N("pomme")),
             NP(D("le"),N("orange")),
             NP(D("le"),N("banane"))),
  VP(V("être"),A("bon")))
```

Coordination

```
S(CP(C("et"),NP(D("le"),N("pomme")),
              NP(D("le"),N("orange")),
              NP(D("le"),N("banane"))),
  VP(V("être"),A("bon")))
```

La pomme, l'orange et la banane sont bonnes.

Coordination

```
S(CP(C("et"),NP(D("le"),N("pomme")),
              NP(D("le"),N("orange")),
              NP(D("le"),N("banane"))),
  VP(V("être"),A("bon")))
```

La pomme, l'orange et la banane sont bonnes.

```
S(CP(C("et"),NP(D("le"),N("pomme"))),
  VP(V("être"),A("bon")))
```

DIRO

rali

Coordination

```
S(CP(C("et"),NP(D("le"),N("pomme")),
             NP(D("le"),N("orange")),
             NP(D("le"),N("banane"))),
  VP(V("être"),A("bon")))
```

La pomme, l'orange et la banane sont bonnes.

```
S(CP(C("et"),NP(D("le"),N("pomme"))),
  VP(V("être"),A("bon")))
```

La pomme est bonne.

Réutilisation d'expression

```
const pomme = NP(D("le"),N("pomme"))
S(Pro("lui").c("nom"),
  CP(C("et"),
     VP(V("manger"),pomme),
     VP(V("aimer"),pomme.pro())))
```

Réutilisation d'expression

```
const pomme = NP(D("le"),N("pomme"))
S(Pro("lui").c("nom"),
  CP(C("et"),
     VP(V("manger"),pomme),
     VP(V("aimer"),pomme.pro())))
```

Il mange la pomme et l'aime.

Réutilisation d'expression

```
const pomme = NP(D("le"),N("pomme"))
S(Pro("lui").c("nom"),
  CP(C("et"),
     VP(V("manger"),pomme),
     VP(V("aimer"),pomme.pro())))
```

Il mange la pomme et l'aime.

```
const pomme = ()=>NP(D("le"),N("pomme"))
S(Pro("lui").c("nom"),
  CP(C("et"),
     VP(V("manger"),pomme()),
     VP(V("aimer"),pomme().pro())))
```

Construction graduelle

```
let pommes = NP(D("un"),N("pomme").n("p"))
pommes.add(A("petit"))
let s = S(Pro("lui").c("nom"),
          VP(V("manger"), pommes))
s.add(Adv("maintenant").a(","),0)
s.toString()
```

Construction graduelle

```
let pommes = NP(D("un"),N("pomme").n("p"))
pommes.add(A("petit"))
let s = S(Pro("lui").c("nom"),
          VP(V("manger"), pommes))
s.add(Adv("maintenant").a(","),0)
s.toString()
```

Maintenant, il mange des petites pommes.

Autres *trucs pratiques*

- Élision, euphonie tenant compte du formatage

```
S(Pro("lui").c("nom"),
  VP(V("manger"),
     NP(D("ce"),N("abricot").tag("b"))))
```

- Écriture des nombres

- Formatage des dates absolues et relatives

DIRO

rali

Autres *trucs pratiques*

- Élision, euphonie tenant compte du formatage

```
S(Pro("lui").c("nom"),
  VP(V("manger"),
     NP(D("ce"),N("abricot").tag("b"))))
```

Il mange cet **abricot**.

- Écriture des nombres

- Formatage des dates absolues et relatives

Autres *trucs pratiques*

- Élision, euphonie tenant compte du formatage

```
S(Pro("lui").c("nom"),
  VP(V("manger"),
     NP(D("ce"),N("abricot").tag("b"))))
```

Il mange cet **abricot**. Il mange cet <b>abricot<b>.

- Écriture des nombres

- Formatage des dates absolues et relatives

Autres *trucs pratiques*

- Élision, euphonie tenant compte du formatage

```
S(Pro("lui").c("nom"),
  VP(V("manger"),
     NP(D("ce"),N("abricot").tag("b"))))
```

Il mange cet **abricot**. Il mange cet <b>abricot<b>.

- Écriture des nombres

```
NP(NO(3).dOpt({nat :true}),N("avion"))
```

- Formatage des dates absolues et relatives

Autres *trucs pratiques*

- Élision, euphonie tenant compte du formatage

```
S(Pro("lui").c("nom"),
  VP(V("manger"),
     NP(D("ce"),N("abricot").tag("b"))))
```

Il mange cet **abricot**. Il mange cet <b>abricot<b>.

- Écriture des nombres

```
NP(NO(3).dOpt({nat :true}),N("avion"))
```
 trois avions

- Formatage des dates absolues et relatives

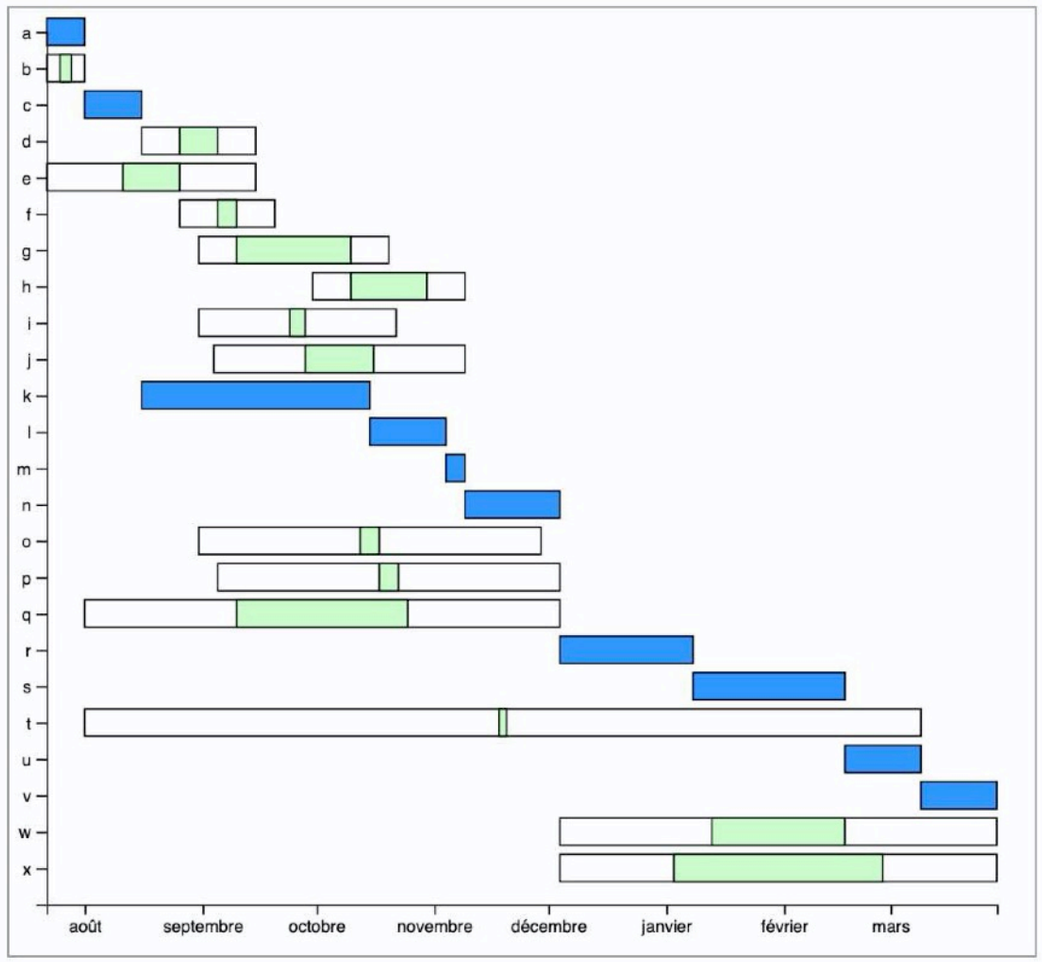
Utilisation dans des pages web

- Démos
  - éditeur d'expression
  - phrase via choix dans des menus
  - toutes les variantes d'une phrase
  - reproduction du Petit Chaperon Rouge
- Jeux de langue
  - Exercices de grammaire et d'orthographe
  - Exercices de style de Queneau
  - Augmentation de Perec
- Data-to-text

Étapes

Le tableau de gauche correspond aux données reçues: à chaque étape sont associés un identificateur, sa durée en nombre de jours ainsi que les identificateurs des tâches qui en dépendent. On peut modifier la date de début du projet ici: `2018-07-22` .

Id	Description	# days	Precedes
a	L'étude du chauffage.	10	c,q,t
b	L'étude de la couverture.	3	c,u
c	L'étude de la charpente.	15	k,d,p
d	L'étude du béton armé.	10	f,o
e	La préparation du terrain.	15	g,i,o,f
f	L'installation du chantier.	5	g,i,o
g	Le terrassement pour les puits.	30	h
h	Le béton pour les puits.	20	n
i	Le terrassement pour les longrines.	4	j
j	Le béton pour les longrines.	18	n
k	La construction de la charpente en atelier.	60	l,m
l	Une première couche de peinture en usine.	20	m,s
m	Le transport de la charpente.	5	n
n	Le montage de la charpente.	25	x,w,r
o	Le terrassement pour les canalisations.	5	p
p	La maçonnerie pour les canalisations.	5	w,r
q	Les approvisionnements pour le chauffage.	45	r
r	La pose du chauffage.	35	s
s	Les autres couches de peinture.	40	u
t	L'étude pour l'électricité.	2	v
u	L'isolation de la couverture.	20	v
v	L'installation de l'électricité.	20	
w	Le dallage.	35	
x	Les enduits.	55	



Le système calcule le chemin critique et ainsi que les dates les plus tôt et le plus tard auxquelles ces étapes peuvent débuter afin de ne pas retarder le projet. Il produit ensuite un texte (réalisé par `jsRealB`) décrivant ces étapes. En passant la souris sur une tâche dans le graphique, apparait une infobulle donnant des informations sur la tâche surlignée en jaune et indiquée en bleu dans le texte et dans le tableau. Les tâches immédiatement préalables et dépendantes sont indiquées par une bordure rouge.

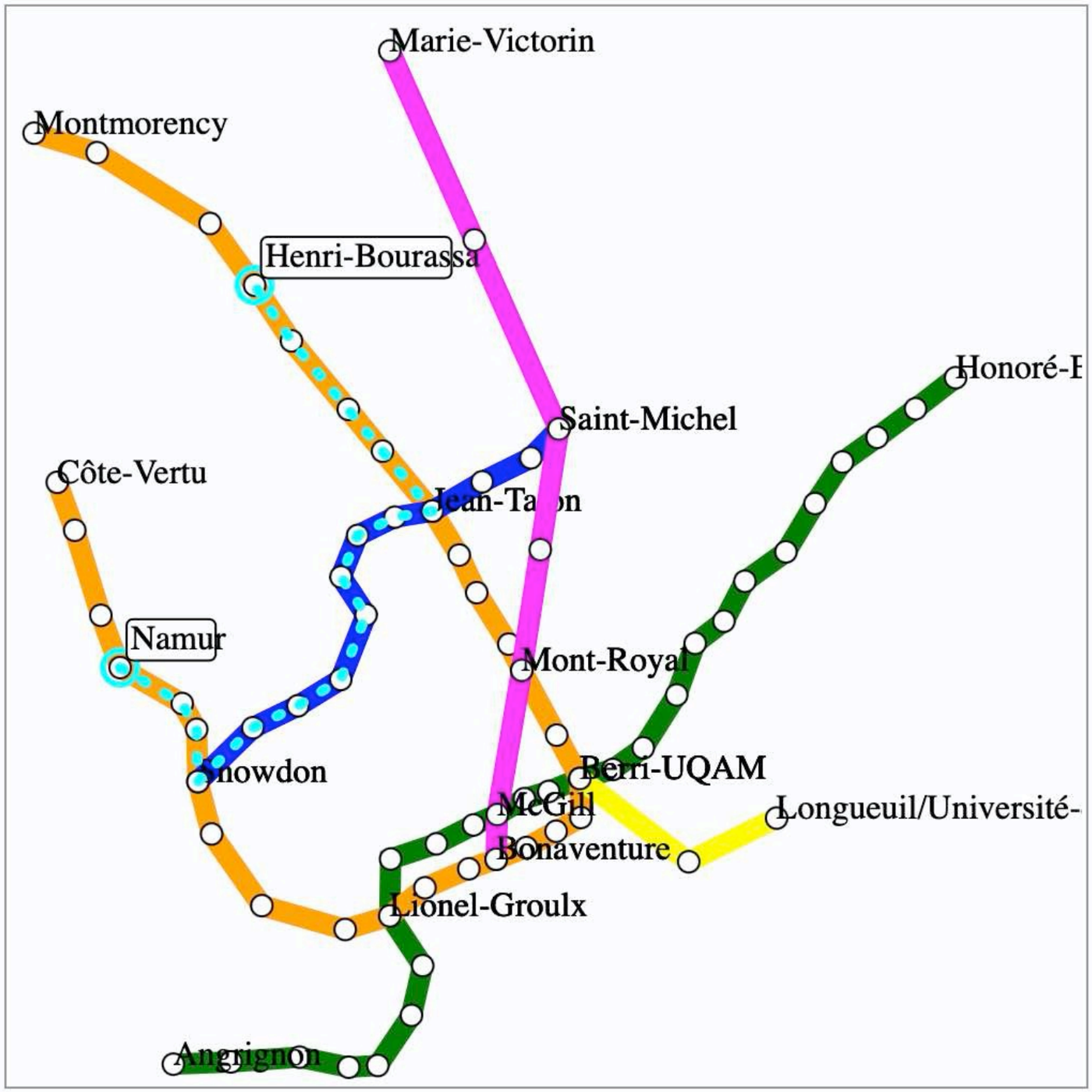
Texte décrivant la suite *optimale* des travaux à partir du 22 juillet 2018

Pour compléter la construction du bâtiment, il faudra au moins deux cent cinquante jours. Il faut commencer par planifier le chauffage, planifier la couverture et préparer le terrain.

Puis il faudra planifier la charpente, s'approvisionner pour le chauffage et planifier l'installation de l'électricité. Le 16 août 2018 marquera le début de l'étude du le béton armé et la construction de la charpente en atelier. Ensuite il faudra installer le chantier. Le 31 août 2018, on devrait s'occuper d'effectuer le terrassement pour les puits, terrasser les longrines et terrasser les canalisations. Le 4 septembre 2018, on devrait s'occuper de couler le béton pour les longrines. Le 5 septembre 2018, on devrait s'occuper de bétonner les canalisations. Ensuite on passera au béton pour les puits. Par la suite, il faudra appliquer une première couche de peinture en usine. Après cela, il faudra transporter la charpente. Par la suite, on passera au montage de la charpente. Le 4 décembre 2018 marquera le début de la pose du le chauffage, le dallage et les enduits. Puis il faudra terminer la peinture. Puis il faudra isoler la couverture.

On finira par l'installation de l'électricité. La construction du le bâtiment sera complétée le 29 mars 2019.

Itinéraire dans le métro



Pour aller de Namur vers Henri-Bourassa

Vous êtes à 48 minutes de votre destination, embarquez sur la ligne orange (direction Montmorency) jusqu'à Snowdon

Passez sur la ligne bleue vers Saint-Michel pour 8 stations.

Votre destination Henri-Bourassa sera la quatrième station sur la ligne orange (direction Montmorency)

DIRO

rali

Évaluation de couverture
60 phrases tirées des UD 2.8 (en, fr)

- moins d'une milliseconde par phrase
- on a pu reproduire toutes les phrases verbatim
- a permis de trouver des erreurs dans les annotations UD

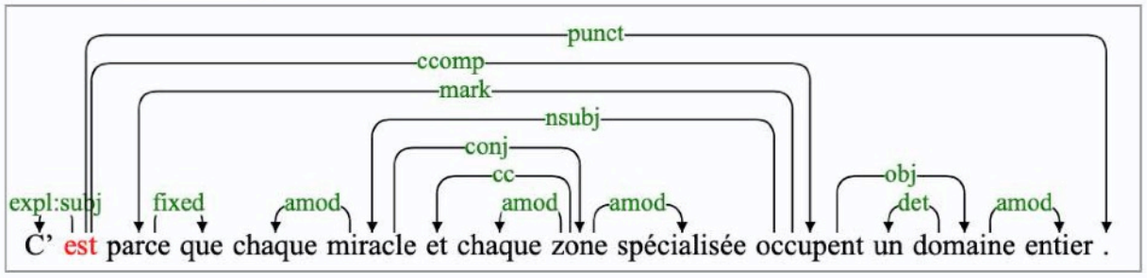
Regénération à partir de Universal Dependencies

Afficher les instructions | Choisir un fichier UD | initialUDs | English version

ID	FORM	LEMMA	UPOS	XPOS	FEATS	HEAD	DEPREL	DEPS	MISC
1	C’	ce	PRON	PDEM	_	2	expl:subj	_	SpaceAfter=No|wordform=c'
2	est	être	VERB	VBC	Mood=Ind|Number=Sing|Person=3|Tense=Pres|Ver…	0	root	_	_
3	parce	parce	ADV	IN	_	11	mark	_	_
4	que	que	SCONJ	IN	_	3	fixed	_	_
5	chaque	chaque	ADJ	JJ	Gender=Masc|Number=Sing	6	amod	_	_
6	miracle	miracle	NOUN	NN	Gender=Masc|Number=Sing	11	nsubj	_	_
7	et	et	CCONJ	CC	_	9	cc	_	_
8	chaque	chaque	ADJ	JJ	Gender=Fem|Number=Sing	9	amod	_	_
9	zone	zone	NOUN	NN	Gender=Fem|Number=Sing	6	conj	_	_
10	spécialisée	spécialisé	ADJ	JJ	Gender=Fem|Number=Sing	9	amod	_	_
11	occupent	occuper	VERB	VBC	Mood=Ind|Number=Plur|Person=3|Tense=Pres|Ver…	2	ccomp	_	_
12	un	un	DET	DT	Gender=Masc|Number=Sing	13	det	_	_
13	domaine	domaine	NOUN	NN	Gender=Masc|Number=Sing	11	obj	_	_
14	entier	entier	ADJ	JJ	Gender=Masc|Number=Sing	13	amod	_	SpaceAfter=No
15	.	.	PUNCT	.	_	2	punct	_	_

Ne montrer que: Différences ☐ Avertissements ☐ Non projectives ☐ | Analyser | 5 phrases | ! C’est parce que chaque miracle et chaque zone spécialisée occupent un domaine entier.

Affichage: Liens | Espacement en pixels: Mot 5 Lettre 0



Ligne	145
sent_id	n02040023
text	C'est parce que chaque miracle et chaque zone spécialisée occupent un domaine entier
TEXT	C'est parce que chaque miracle et chaque zone spécialisée occupent un domaine entier 3 avertissements: • Adv('parce':: absent du lexique français. • A('chaque':: absent du lexique français mais existe comme D. • A('chaque':: absent du lexique français mais existe comme D.
	aucune différence

Masquer l'éditeur jsRealB

```
S(Pro("ce").c("nom"),
  VP(V("être").t("p").pe("3").n("s"),
     S(Adv("parce"),
       C("que"),
       CP(C("et"),
          NP(A("chaque").pos("pre").g("m").n("s"),
             N("miracle").g("m").n("s")),
```

Autres utilisations

- localisation des messages d'erreur de jsRealB
- module *node.js* disponible sur *npm*
  - génération de corpus d'apprentissage via les variantes
    - pour un système de réponse à des questions
    - pour apprendre à traiter des négations
- serveur jsRealB (traite aussi le JSON)
- génération à partir d'AMR
  - SWI-Prolog
  - Python

pyrealb : version Python

- formalisme syntaxique *identique* à jsRealB
- mêmes possibilités que jsRealB
- réalisation via `str(...)`

```
>>> from pyrealb.all import *
>>> loadFr()

>>> print(S(Pro("lui").c("nom"),
...   VP(V("donner").t("pc"),
...     NP(D("un"), N("pomme")).pro())
... ).typ({"neg":true,"pas":true}))

Elle n'a pas été donnée par lui.
```

DIRO

rali

Génération de bulletins météo

Données JSON


```
{"header":["regular","2019-12-12T11:30:00","next","2019-12-12T15:45:00"],
 "names-en":["Gaspésie National Park – Murdochville"],
 "names-fr":["Parc national de la Gaspésie – Murdochville"],
 "precipitation-type":[[-51,-48,"snow",...
                       [151,163,"flurries"]],
 "precipitation-accumulation":[[-51,-48,"snow",8.0],...
                               [97,100,"snow",1.0]],
 "precipitation-probability":[[-67,-51,0],...
                              [199,223,10]],
 "sky-cover":[[-67,-51,8,8],
              [-51,-32,8,8],...
              [166,223,2,2]],
 "temperatures":[[-56,-53,4],
                 [-53,-41,7],...
                 [220,223,-12]],
 "uv-index":[[-37,-35,0.8],
             [-13,-11,0.8],...
             [35,37,0.8]],
 "wind":[[-56,-51,"s","speed",15],...
         [196,223,"nw","speed",15]],
 "id":"fpcn74-2019-12-12-1630-r74.7"}
```

Génération de bulletins météo

Sortie

```
WEATHER BULLETIN: regular
Forecasts issued by jsRealB on Thursday, December 12, 2019 at 11:30 a.m. for today and tomorrow.
The next scheduled forecasts will be issued on Thursday, December 12, 2019 at 3:45 p.m.
Gaspésie National Park – Murdochville

Today : Mainly cloudy. 60 percent chance of flurries. Wind west 30
 km/h gusting to 50. High minus 12. Low minus 15.
Tonight : Partly cloudy.  Wind west 15 km/h in the evening. Low minus
 18, with temperature rising to minus 15 by morning.
Friday : Mainly cloudy. Flurries beginning in the morning, amount 5
 cm. Wind southwest 15 km/h in the morning. High minus 7. Low minus
 17.
END

BULLETIN MÉTÉOROLOGIQUE: régulier
Prévisions émises par jsRealB le jeudi 12 décembre 2019 à 11 h 30 pour aujourd'hui et demain.
Les prochaines prévisions seront émises le jeudi 12 décembre 2019 à 15 h 45.
Parc national de la Gaspésie – Murdochville

Aujourd'hui : Généralement nuageux. 60 pour cent de probabilité
 d'averses de neige. Vents de l'ouest de 30 km/h avec rafales à 50.
 Maximum moins 12. Minimum moins 15.
Ce soir et cette nuit : Partiellement couvert.  Vents de l'ouest de 15
 km/h dans la soirée. Minimum moins 18, températures à la hausse pour
 atteindre moins 15 en matinée.
Vendredi : Généralement nuageux. Averses de neige débutant le matin,
 accumulation de 5 cm. Vents du sud-ouest de 15 km/h le matin. Maximum
 moins 7. Minimum moins 17.
FIN
```

Conclusion

- jsRealB / pyrealb : rapides et prévisibles
- Intégration
  - jsRealB : pour les pages web
  - pyrealb : pour intégration à des programmes Python
- Testés dans plusieurs contextes
- Librement accessibles (sources et démos)

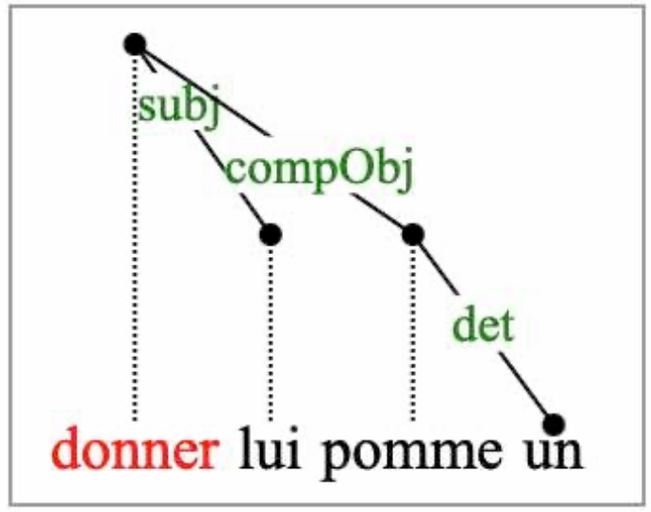
Remerciements

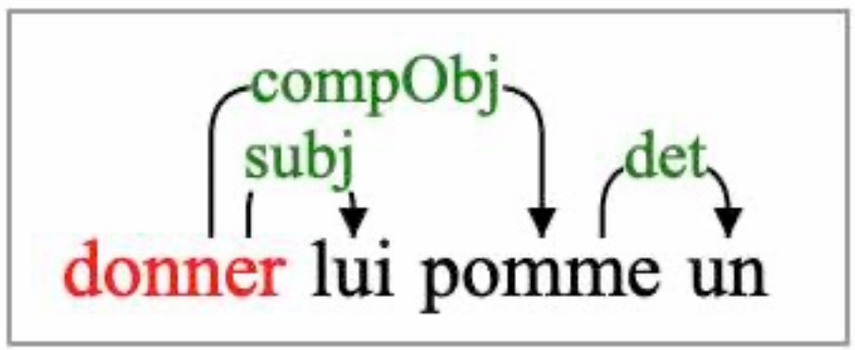
- Nicolas Daoust, MSc 2013 [Français]
- Paul Molins, stage 2015 [Bilingue]
- Francis Gauthier, stage 2015 [variantes]
- Fabrizio Gotti, RALI
- François Lareau, OLST
- Alessandro Sordoni, Microsoft

Travaux futurs

- Entrée sous forme de structure de dépendances

```
root(V("donner").t("pc"),
     subj(Pro("lui").c("nom")),
     compObj(N("pomme"),
             det(D("un")).pro())
).typ({"neg":true,"pas":true})
```





- Combinaison avec méthodes d'apprentissage

Références

- Guy Lapalme, *The jsRealB Text Realizer: Organization and Use Cases,* jan 2021, https://arxiv.org/pdf/2012.15425.pdf
- G. Lapalme. *Validation of Universal Dependencies by regeneration*. 5th Universal Dependencies Workshop, 10p., Sofia, Bulgaria, March 2022. SyntaxFest 2021: 6th International Conference on Dependency Linguistics.
- Guy Lapalme. *Realizing Universal Dependencies structures using a symbolic approach*. Proceedings of the 2nd Workshop on Multilingual Surface Realisation (MSR), (EMNLP-2019), 8 pages, Hong-Kong, nov 2019. ACL.
- Guy Lapalme. Verbalizing AMR structures. Aug 2019.

- Paul Molins and Guy Lapalme. *JSrealB: A bilingual text realizer for web programming*. In Proceedings of the 15th European Workshop on Natural Language Generation (ENLG), pages 109–111, Brighton, UK, September 2015. Association for Computational Linguistics.
- Nicolas Daoust and Guy Lapalme. *JSreal: A Text Realizer for Web Programming*. In Nuria Gala, Reinhard Rapp, and Gemma Bel-Enguix, editors, Language Production, Cognition, and the Lexicon, number 6636 in Text, Speech and Language Technology, chapter 21, pages 363–378. Springer, jul 2014.

Liens

- jsRealB
  - page au RALI
  - GitHub
  - Documentation
  - Environnement de développement
  - Tutoriel
  - Notebook Observable
  - Démonstrations
- pyrealb
  - GitHub