	GINsim	Libddd	Pint	Méthode de Rocca	Biocham	ASP-PH
Abréviation	Gène Interaction Network Simulation	Library of Data Decision Diagrams	Process hitting Tools	------		------
Approche	Un logiciel d'analyse et de simulation de réseaux d'interactions génétiques (interface graphique + simulation + un toolbox pour l'analyse graphique)	Une librairie pour la manipulation des diagrammes de décision des données et les diagrammes hiérarchiques de décision. Lib-its : libraire de model-checking (LTL, CTL et atteignabilité)	Vérification des propriétés dynamiques pour des réseaux de régulation biologiques présentés en Process Hitting	Vérification des propriétés du modèle checking pour des réseaux d'états de transition		Recherche exhaustive des propriétés dynamiques d'un rèseau en PH
Outils/ Langage	- Logiciel développé en Java,	- Librairies développées en C++	- SAT, Clasp - Process Hitting	ASP Graphes d'états de transition		ASP
Modèle en entrée	- Graphe de régulation logique (R. Thomas) les fichiers de ces graphes sont d'extension « .zginml » et en format XML. - Graphe de transition (synchrone/ asynchrone/ mixé)	- Instantiable Transition Systems (fichier xml)	PH	RT	NuSMV	Réseau PH écrit en ASP
Méthode	Il prend en entrée le réseau de régulation logique et le traduit en un graphe de transition en calculant l'évolution possible du réseau pour donner les états stables et résoudre le problème de l'atteignabilité	- Its-reach : calculer les états atteignables - **its-ctl** : vérification des propriétés CTL (EF, AF, EG..) - its-ltl : vérification des propriétés LTL	Calculer et retourner les points fixes du réseau Une sous-approximation pour la vérification de l'atteignabilité	Il analyse le réseau d'états de transition selon des propriétés choisies et affiche tous les états qui puissent aboutir à vérifier ces propriétés.		Faire évoluer le réseau en vérifiant si les processus cibles sont atteints
Points communs avec notre approche	- Choix des états initiaux et des états cibles. - Calculer les évolutions possibles du réseau (réseau de 663.552 états en 57.15s) - Calculer et afficher les états stables - Calculer et afficher le chemin entre 2 états	-	Utilisation du même formalisme : le PH.	Utilisation d'ASP		---------
Points différents avec notre approche	-Le chemin affiché est une succession d'états du graphe d'états de transitions et non pas juste les changements effectués. - Pouvoir d'analyser des circuits élémentaires	Utilisation de ITS et nous du PH Réponse « True » ou « False »	Notre approche est exacte par rapport à la leur qui est basée sur l'approximation pour l'atteignabilté	Leur résultat est l'ensemble des états initiaux et nous on affiche le chemin.		----------
Spécification	Interface graphique très claire pour la manipulation et la simulation des graphes	Traitement de tous les propriétés CTL et LTL	La réponse est assez rapide	Ils traitent toutes les propriétés CTL et LTL		Chemin exact et le plus court et une réponse relativement rapide

Inconvénients	Ne peut pas afficher les graphes de transition de grande taille (~17.470 états)	Gourmant en temps et mémoire	Ne retourne pas le chemin pour atteindre l'objectif mais seulement une réponse (Vrai/Faux)	Le résultat prend plus de temps pour s'afficher.		Parfois inconclusive (prévoir le nombre d'étapes)
Etude de cas identiques : ERBB G1-S avec un état cible (ensemble de cibles)	- taille du graphe: 1 048 576 - réponse retournée : chemin d'activation - temps de calcul: 2m01.64s pour tout le graphe - temps mis pour trouver le chemin est presque instantanée - 1 état stable: calculé instantanément	- réponse retournée: True - temps mis pour répondre: 1m55.385s (user : 1m57.315s / sys : 0m1.084s)	- réponse retournée: True - temps mis pour répondre: 0m0.007s			- taille du graphe: 20 composants - réponse retournée : chemin d'activation - temps de calcul: 0m11.840 s (prepare: 1.010 / prepro: 0.810 / solving: 9.860) pour 18 étapes - nombre de chemins possibles: 1260 - 3 états stables: calculés en 0.010 s (instantanément)
ERBB G1-S avec une seule cible	Impossible d'avoir qu'un seul cible : il faut donner un état initial et un état cible et puis chercher le chemin	- réponse retournée: True - temps mis pour répondre: 1m54.961s (user: 1m53.924s, sys: 0m.1.052s)	- réponse retournée: True - temps mis pour répondre: 0m0.004s			- temps de calcul: 5.020 s (prepare: 1.010 / prepro: 0.850 / solving: 3.120) pour 18 étapes - nombre de chemins possibles: 1260
TCRsig40	- taille du graphe: 1.206.984 - temps de calcul: ~3mn 01.64s pour tout le graphe - pas de chemin trouvé (peut être le graphe est incomplet ou il faut définir/deviner l'état objectif (tous les niveaux de tous les composants) - 5 états stables: calculés instantanément	- réponse retournée: aucune réponse - temps mis pour répondre: 10m57.292s (user: 10m36.593s, sys: 0m.5.019s)	- réponse retournée: True - temps mis pour répondre: 0m0.004s			- taille du graphe: 40 composants (50 avec les sortes coopératives) - temps de calcul du chemin: 0m95.080 s = ~1m.584 (prepare: 1.650 / prepro: 1.320 / solving: 91.540s) pour 25 étapes - nombre de chemins possibles: 103.296+ (calculés en 519.550s) - 7 états stables: calculés en 0.000 s (instantanément)