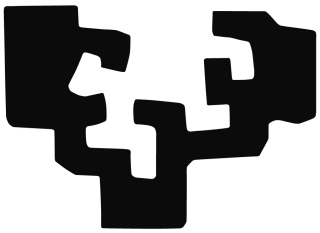
Euskal Herriko Unibertsitatea



Informatika Ingeniaritzako Gradua

Konputazioa

Gradu Amaierako Proiektua

Polimerizazio-prozesu Simulazioaren Atzeranzko Ingeniaritza: Erreakzio-baldintzen Optimizazioa

Egilea:
Aitor Alkorta

Zuzendariak:
Borja Calvo
Christian Blum

2016(e)ko ekainaren 24a

LABURPENA

Polimeroak polimerizazio-prozesu batean erreakzio-baldintza batzuk ezarrita (tenperatura, presioa, monomeroak, etab.) eratzen diren materialak dira. Eskaturiko propietateak dituzten polimeroak bilatzea erronka handia da industrian, polimerizazioaren simulazioan lorturiko emaitzak zein izango diren aurreikustea oso konplexua baita.

Nahi den polimero mota lortzeko, polimerizazio-prozesuetako erreakzio-baldintzak optimizatu behar dira. Ez bada teknika espezifikorik erabiltzen, propietate jakin bateko polimero bat lortzeko sistematikoki simulatzailearen proba-errore bidezko exekuzio asko egin behar dira. Horrek, denbora eta konputazio kostu handiak dakartza.

Prozesua azkartzeko, proiektu honetan bilaketa heuristikoak (algoritmo metaheuristikoak) erabiliko dira, algoritmo horiek konputazio ahalmen mugatuak daudenean eta lortuko diren emaitzak aurreikusi ezin direnean soluzio onak lortzeko erakusten duten eraginkortasunagatik.

Proiektuan, erreakzio-baldintzen optimizazio problema aztertzeko etilenotik polietileno tereftalatoa (PET) lortzeko prozesuaren simulatzaile bat erabili da. Lehenik, soluzioen espazio aztertu da simulatzailea analizatuz. Ondoren, algoritmo metaheuristiko batzuk inplementatu dira eta problema mota honetarako egokienak zein izan daitezkeen aztertu da.

Guzti hori egin ondoren, antzeko proiektu bat garatzean izan daitezkeen arazoak identifikatu eta optimizazio-algoritmo hobeak inplementatzeko ideiak eman dira, proiektu honetan eginikoa lagungarria izango delakoan.

GAIEN AURKIBIDEA

IRUDIEN ZERRENDA

TAULEN ZERRENDA

ALGORITMOEN ZERRENDA

1 SARRERA

Polimeroak gaur egun gehien erabiltzen diren materialetakoak dira. Izan ere, lehengaiak lortzeko prozesuak merkeak dira eta ezaugarri berdintsuak dituzten beste material batzuen ordez erabil daitezke. Polimero ezagunenetan, poliuretanoa (PU), poliestirenoa (PS), polibinil kloruroa (PVC), polietilenoa (PE) eta polietileno tereftalatoa (PET) ditugu.

Polimeroak molekula mota ezberdinez eginak daude. Molekula horiek monomero izena dute eta monomeroak kateatzeko prozesu kimikoak *polimerizazio-prozesua*. Polimerizazioan sortzen diren kateek polimero dute izena. Polimeroak sortzean hainbat faktorek parte hartzen dute, hala nola, hasarazleek, monomeroek, tenperatura edo presioa. Horien bitartez, partikulen arteko *erreakzio-baldintzak* zein diren zehazten da.

Simulatzaile batek sarrerako parametro batzuk ezarrita, mundu errealean gertatzen den polimerizazio-prozesu baten portaera simulatzen du emaitza bat itzuliz, luzera ezberdineko polimero-kateen distribuzio bat hain zuzen. Simulatzaileek emaitza egokia eman dezaten, eredu matematiko egoki bat erabiltzeaz gain, hainbat faktoreek eragindako aldaketak kontuan hartu behar dituzte. Bi eredu matematiko mota daude, deterministak, ekuazio diferentzialetan oinarriturikoak eta estokastikoak, *Monte Carlo* motako tekniken bidez ebazten direnak [Mohammadi et al., 2014] [Hamzehlou et al., 2014]. Monte Carlo-n oinarritutako teknikak oso erabiliak dira gaur egungo simulatzaileetan, errore oso txikiarekin kostu handiko adierazpen matematikoen hurbilketak lortzen dituztelako. Horrela, konplexutasun eta kostu txikiagoak dituzten simulazioak egitea ahalbidetzen dute, baina hurbilketak direnez ez dira guztiz zehatzak izaten simulazioen emaitzak.

Bestalde, propietate ezberdinak dituzten polimero berriak sortzeko zailtasuna da polimeroen industriako arazoetako bat. Arazo horri aurre egiteko, batetik, simulatzaile zehatz bat behar da eta, bestetik, nahi den propietateak dituen polimeroa lortzeko *atzeranzko ingeniaritza* erabili behar da, simulatzaileari eman behar zaizkion erreakzio-baldintzak zein diren jakiteko. Simulatzaileak askotan konplexuak izaten direnez, ezin da aurreikusi lortuko den emaitza, hots, ezin zaie eredu matematikoei buelta eman.

Emaitzarik onena lortzeko, polimerizazio-prozesua simulatu behar da parametro konbinazio posible guztiak aztertuz. Konbinazio posible horiei guztiei *soluzio bideragarrien espazioa* edo *bilaketa espazioa* deritzo. Bilaketa espazio osoa aztertzeak kostu handiak ditu, bai denborarekiko eta baita konputazio ahalmen aldetik ere. Beste hitzetan, azterketa hori konputaezina da.

Soluzioen kalitatea neurtzerik balego, *optimizazio-algoritmoak* erabil daitezke problema konplexuetan. Optimizazio problemak ebazteko formalizatu egin behar dira, eta

horretarako soluzioen kodeketa, murrizketak eta *helburu-funtzioa* definitu behar dira. Helburu-funtzioak lortu den emaitzaren kalitatea neurtzen du eta bilaketa espazioko emaitza onenari *soluzio optimo globala* deritzo. Helburu-funtzioa maximizatu edo minimizatu egin daiteke problema motaren arabera.

Lortu nahi dugun polimeroaren ezaugarriak simulatzaileak itzultzen duenarekin aldera dezakegunez, erreakzio-baldintza optimoak bilatzea optimizazio problema bat da.

Bilaketa espazio osoa aztertzea konputaezina delako, sarrera berdinekin lorturiko emaitzak aldakorrak direlako edota ebaluazioetan denbora mugak jarri behar direlako, hainbat optimizazio problema ezin dira ebatzi, hau da, soluzio optimo globala lortzea ezin da bermatu. Horrelakoetan, *bilaketa heuristikoak* (*algoritmo metauristikoak*) [Blum and Roli, 2003] erabiltzen dira. Algoritmo metaheuristikoak intuizioan oinarritzen diren algoritmoak dira, eta optimoa ez bermatu arren, soluzio onak lortzen dituzte normalean.

1.1 AURREKARIAK

Gaur egun, ohikoa bihurtu da simulatzaileen atzeranzko ingeniaritzaren erabilera parametroen optimizazioan. [Gaillard et al., 2009] artikuluan atzeranzko ingeniaritza erabiltzen dute agenteetan oinarrituriko ereduak analizatzeko eta hobetzeko. [García et al., 2008] artikuluaren egileek industriako piezen estrukturak optimizatzeko, algoritmo genetiko bat proposatzen dute. Optimizazio algoritmoen analisi orokorrago bat egiten dute [Hemker, 2010] eta [Rios and Sahinidis, 2013] artikuluetan. Lehenengoan, atzeranzko ingeniaritza erabiliz deribaturik gabeko optimizazio ezberdin askoren analisia egiten dute, problema mota ezberdinetan algoritmoek izan ditzaketen alde onak eta txarrak aztertuz. Bigarrenean, ingeniaritzako hainbat arlotako problemetan aplikatu eta lorturiko emaitzak argitara eman dituzte.

Proiektuan aztertuko den polimerizazioko erreakzio-baldintzen optimizazioari dagokionez, azken ikerkuntza garrantzitsuenak sare neuronalen erabilera eginak dira [Fernandes et al., 2004] [Nascimento et al., 2000]. Ikerketa horietan sare neuronalek polimeroen industrian duten aplikazioaren garrantzia nabarmentzen dute eta atzeranzko ingeniaritza erabiliz, sare neuronalek erreakzio-baldintzen optimizazioa egiteko dituzten abantailak adierazten dituzte.

1.2 PROIEKTUAREN HELBURUA

Polimerizazio-prozesu simulatzaile sinple bat erabilita, nahi den polimero mota lortzeko erreakzio-baldintzen optimizazioak dakartzan problemaren ezaugarriak aztertu nahi dira proiektuan.

Polimerizazio simulatzaileari sarrera jakin batzuk emanez, irteeran itzuliko duen emaitza ezin daiteke aurreikusi, optimizatu nahi den funtzioa ez baita ezaguna. Gainera, simulatzaileak eredu probabilistiko bat balia dezake eta, beraz, beti egon daitezke aldaketak emaitzetan, baita parametro sarrera berak behin baino gehiagotan finkatuta ere. Honen ondorioz, optimizazioan metodo numerikoen erabilera ez da egokia izango, simulatzailearen funtzioa ezagutzeak problemaren ulermen eta konplexutasun maila oso altuak eskatzen baititu. Beraz, problemaren ebazpenean konplexutasun eta ulermen maila txikiagoak behar dituzten metodo batzuk aztertuko dira.

Proiektuan denbora maximo bat edukiko da soluzio bat lortzeko eta soluzioen bilaketa espazioa handia da. Problema konplexua denez, soluzio optimo global bat lortzea ezin da bermatu. Horregatik, bilaketa espazioa modu eraginkorrean aztertzeko eta emaitza onak lortzeko, algoritmo metaheuristikoen [Blum and Roli, 2003] erabilera egingo da.

Aurreko guztiagatik, proiektuak helburu hauek beteko ditu:

- Simulatzaile sinple bat lortu, diru kosturik eta konputazio ahalmen handirik behar ez duena, gure proiektura egokitzeko.
- Bilaketa espazioaren azterketa egitea, simulatzailearen alde onak eta txarrak ikusteko.
- Algoritmo metaheuristiko mota ezberdinak inplementatu eta hauen funtzionamenduaren analisia egin, erreakzio-baldintzen optimizaziorako duten egokitasuna aztertuz.
- Aztertutako guztiaren ondorioak atera proiektuan lortu dena azalduz.

1.3 DOKUMENTUAREN EGITURA

Lehenengo kapituluan, aurrean dugun problemaren sarrera bat egin da, aurretik egin diren ikerketei gainbegiratu bat eman eta proiektuan lortu nahi diren helburuak azaldu dira.

Bigarren kapituluan proiektuaren helburuak betetzeko proiektuan egin den plangintza azaltzen da.

Hirugarren kapituluan, aukeratu den simulatzailea simulatzailea, problema nola ebatzi den eta erabili diren optimizazio algoritmoak azaldu dira.

Laugarren kapituluan, bilaketa espazioaren eta inplementatuko algoritmoen analisiak egin dira, proba ezberdinak exekutatuz.

Azkenik, bostgarren kapituluan, proiektuan eginiko lanetik ateratako ondorioak azalduko dira identifikaturiko arazoak azalduz eta etorkizunean egin daitekeenari buruz hitz eginez.

2 PLANGINTZA

Kapitulu honetan proiektuan jarraitu den plangintza azaltzen da. Bertan, proiektuaren helburuak gauzatzeko eginiko atazen banaketa, kudeaketa eta antolaketa azalduko dira. Lehenik, atazen banaketa azalduko da, eta bertan atazen mugarriak eta beharko diren baliabideak azalduko dira. Ondoren, *Gantt* diagrama irudikatuko da eta arriskuen identifikazioa eta kontrola azalduko da. Azkenik, proiektuko jarraipen eta kontrola egingo da. Proiektuak, 12 kredituren balioa du, eta kreditu bakoitzak 25 orduko balioa duenez 300 orduko lana planifikatu da.

Taldeko kideak

Proiektuak hiru partaide ditu, bi zuzendari eta proiektuaren egilea. Bi zuzendariak proiektua gainbegiratzeaz arduratuko dira eta proiektuaren egilea, izenak dioen moduan proiektua gauzatzeaz arduratuko da.

2.1 ATAZEN BANAKETA

Proiektua denboran zehar antolaturiko 6 atazetan banatu da. Ataza bakoitzean, honen mugarriak, atazaren azalpena eta behar diren denbora eta baliabideak azalduko dira.

A0. Proiektuaren Kudeaketa eta Gainbegiratzea

Proiektu zuzendarien eta egilearen arteko bilerak periodikoki egingo dira, eginiko lana berrikusiz. Eginiko lana *Bitbucket* plataformara igoko da eta bertatik egingo da gainbegiratzearen zati bat.

Irismena: Proiektu osoa.

Denbora Gastua: 40 ordu.

Baliabideak: Zuzendariak + Bitbucket.

A1. Aurrekariak eta Problemaren Familiarizazioa

Problema honen aurrekariak aztertzea eta problemaren ebazpenerako erabili beharreko tresna, polimerizazio-prozesua eta optimizazio-algoritmoekin familiarizatu.

Irismena: 2015/10/01 - 2016/01/11.

Denbora Gastua: 30 ordu.

Baliabideak: Egilea + ordenagailua.

A2. Simulatzailearen Aukeraketa

Erabili daitezkeen tresnak aztertu ondoren, proiektuan erabiliko den simulatzailearen aukeraketa egin eta simulatzailearen bilaketa espazioa analizatu.

Irismena: 2016/01/11 - 2016/02/22.

Denbora Gastua: 30 ordu.

Baliabideak: Egilea + ordenagailua.

Emangarria: Simulatzailearen aurkezpen bat.

A3. Problemaren Formalizazioa

Optimizazio problema formalizatzeko, soluzioaren kodeketa eta helburu-funtzioak definitu.

Irismena: 2016/02/22 - 2016/03/07.

Denbora Gastua: 20 ordu.

Baliabideak: Egilea + ordenagailua.

Emangarriak: Formalizazioa eta helburu-funtzioaren dokumentua.

A4. Optimizazio-Algoritmoak

Proiektuan erabiliko diren optimizazio-algoritmoak inplementatu.

Irismena: 2016/03/07 - 2016/03/28.

Denbora Gastua: 40 ordu.

Baliabideak: Egilea + ordenagailua.

Emangarriak: Algoritmoen kodea.

A5. Instantzien Definizioa eta Esperimentazioa

Probak egiteko instantziak definitu. Algoritmoekin probak egin, problemaren ebazpenerako dituzten abantailak eta desabantailak zein diren ikusiz. Hau guztia dokumentatu, grafiko eta ondorioak ateraz.

Irismena: 2016/03/28 - 2016/06/06.

Denbora Gastua: 60 ordu.

Baliabideak: Egilea + ordenagailua.

Emangarriak: Dokumentua ateratako grafiko eta analisiekin.

A6. Dokumentazioa

Proiektuan zehar eginikoa dokumentatu, memoria eginez. Proiektuaren aurkezpena prestatu.

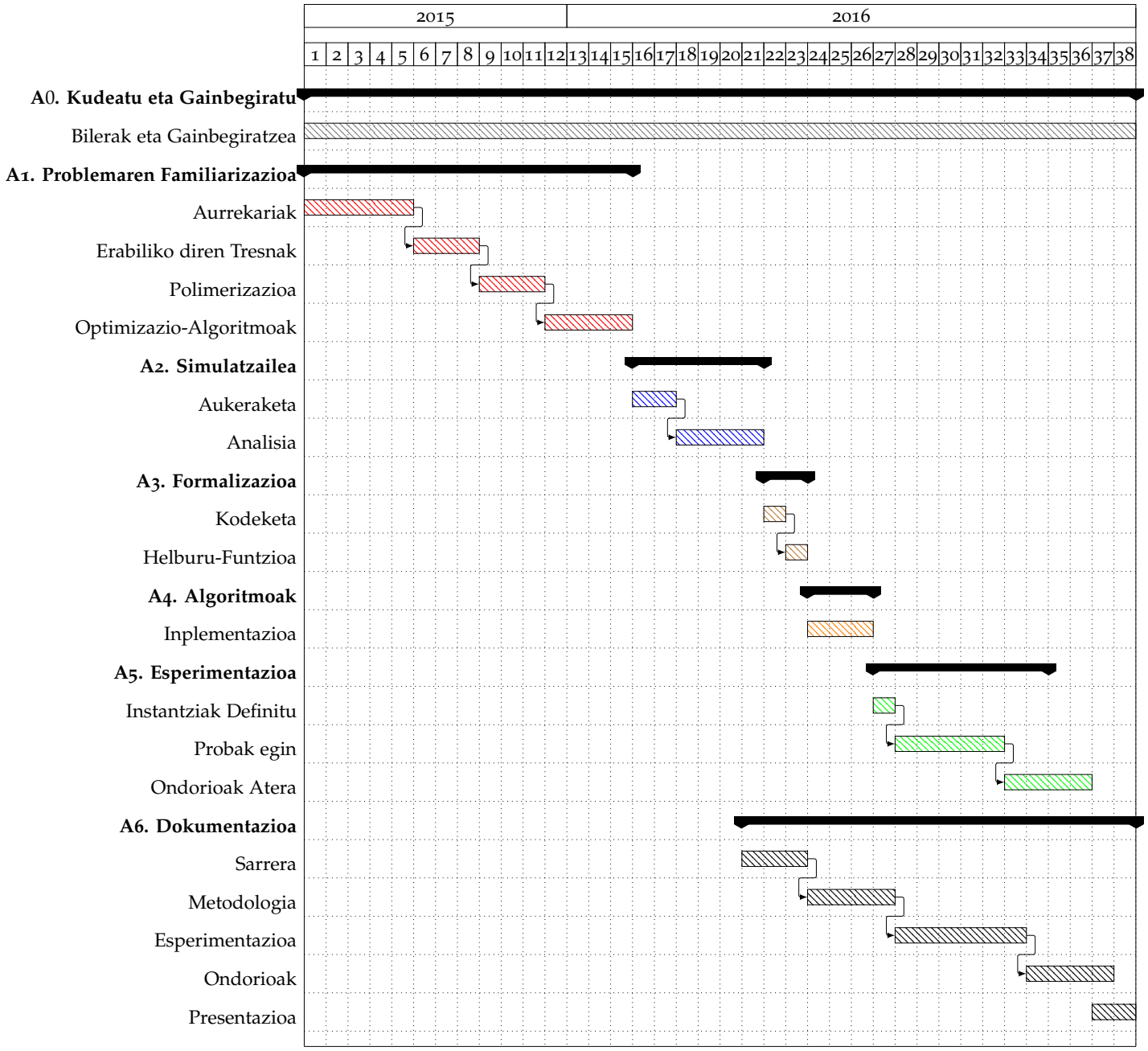
Irismena: 2016/03/14 - 2016/06/20.

Denbora Gastua: 60 ordu.

Baliabideak: Egilea + ordenagailua.

Emangarriak: Memoria eta Aurkezpena.

2.2 GANTT DIAGRAMA



2.3 ARRISKUEN IDENTIFIKAZIOA ETA KUDEAKETA

Atal honetan proiektuarentzako arrisku plana zehaztuko da. Hiru lerro nagusi finkatu dira: Proiektuak atzerapenak izatea, informazio galera eta simulatzaile egokiaren falta. Hauen deskribapena, probabilitatea, inpaktua eta mitigazio neurriak azalduko dira ondoren.

2.3.1 Proiektuaren Atzerapenak

Emangarrietan atzerapenak gerta daitezke eta horrek plangintza osoaren atzerapenak ekar ditzake. Hau ez gertatzeko planifikazioa malgua egin da, atzerapena handia izanda ere planifikazioaren atzerapenik gerta ez dadin. Atzerapenak gertatzeko probabilitatea beraz, baxua kontsideratuko da.

2.3.2 Informazioaren Galerak

Gaur egun informazio guztia ordenagailuetan gordetzen denez, informazio galerak izatea ohikoa da.

Proiektuan informazio galerarik egon ez dadin, hiru neurri hartu dira:

1. Git-en erabilera egin da, Bitbucket plataformara informazioa igoz. Git bertsio kontrolerako sistema bat da, beraz, aldaketa kopuru garrantzitsu bat egiten den bakoitzean, Bitbucket plataformara igoko da informazio guztia.
2. Google Drive-era igoko da momentuan eginiko modifikazio bakoitza.
3. Segurtasun kopia bat egingo da periodikoki aparteko disko gogor batean informazio guztia gordez.

Harturiko neurriekin, informazio galera gertatzeko probabilitatea oso baxua izango da, baina galera gertatuko balitz inpaktua oso handia izango litzateke.

2.3.3 Simulatzailea

Proiektua gauzatzeko simulatzaile baten beharra dago. Simulatzaile egoki bat bilatzea ez da erraza izango eta simulatzaile egoki bat ez topatzea probabilitate handikoa izango da. Hau gertatuko balitz, simulatzailea inplementatu beharko da eta honek planifikazioa atzeratuko luke.

Horrelako atzerapenak ekiditeko planifikazioan 20 orduko denbora tarte bat erreserbatu da eta simulatzailea inplementatzeko nahikoa dela estimatu da.

2.4 PROIEKTUAREN JARRAIPEN ETA KONTROLA

Atal honetan atazak betetzeko eginiko jarraipena azalduko da.

A0. Proiektuaren Kudeaketa eta Gainbegiratzea

Lehen ataza hau, bilerak egitea eta proiektua gainbegiratzea izan denez, ez du konplikazio izan eta astean behin edo bitan egin dira bilerak eta gainbegiratzeak. 40 ordu zeuden planifikatuta baina 30 bat ordu erabili dira guztira.

A1. Aurrekariak eta Problemaren Familiarizazioa

Ataza honetan ez da arazorik izan eta planifikatu zen moduan 25 ordu inguruko iraupen totala izan du eta 2015/12 datarako amaitua zegoen.

A2. Simulatzailearen Aukeraketa

Zati hau izanda konplikazio gehien izan dituena. Simulatzailearen aukeraketa uste baino konplexuagoa izan da eta ez denez proiekturako guztiz egokia zen ezer bilatu, simulatzaile berri bat inplementatu behar izan da. Ondorioz, 30 orduko gastu estimatua 45 ordutara igo da. Arriskuen kudeaketan 20 orduko tarte bat erreserbaturik zegoenez atzerapenetarako, planifikazioak ez du atzerapenik izan.

A3. Problemaren Formalizazioa

Helburu-funtzio ezberdinak probatu diren arren problemaren formalizazioa mugarrien barruan egin da eta 20 ordu inguruko iraupena izan du.

A4. Optimizazio-Algoritmoak

A1 atazan azterturiko hainbat optimizazio-algoritmo aztertu ondoren, Ausazko Bilaketa, Line Search eta Algoritmo Genetikoa algoritmoak inplementatu dira. Ez da konplikaziorik izan eta aurreikusitako iraupena baino 20 ordu gutxiagoan egin da.

A5. Instantzien Definizioa eta Esperimentazioa

Instantzien definizioak eta algoritmoen esperimentazioak luze jo dute, lorturiko emaitzak askotan ez baitziren esperotakoak eta ondorioz bilaketa espazioaren proba kopuru gehiago egin behar izan dira. Horrek, 90 ordura igo du atazaren denbora gastua eta atzerapenak sortu dira planifikazioan.

Ataza	Planifikatutakoa	Erreala
*A*0	40	30
*A*1	30	25
*A*2	30	45
*A*3	20	20
*A*4	40	20
*A*5	60	90
*A*6	60	90
Erretserbatuta	20	
Guztira	300	320

1. taula: Proiektuan planifikaturiko gastuak vs errealitatean eginikoak (ordutan)

A6. Dokumentazioa

Dokumentazioa proiektuaren irismenaren erdialdetik aurrera egiten hastea planifikatuta egon arren, lehenago hastea erabaki da A2 atazan lorturiko analisien emaitzak eta aurrekariak dokumentatzen hasiz. Gainera, 90 orduko iraupena izan du eta ez 70 ordukoa planifikatu zen bezala.

Guzti hau laburbilduz, sartu beharreko ordu kopuru totala 300 ordu ziren eta 320 inguru sartu dira guztira. Espero ez ziren atzerapenak, A5 eta A6 atazetan izan dira (A2 atazean ere atzerapena izan da baina hori esperotako atzerapena zen), baina nahiko modu malguan egina zegoenez planifikazioa ez da arazo handirik egon mugarriekin eta proiektua entrega egunerako bukatu ahal izan da. Ikusi 1 taula sartutako denborak ikusteko.

3 METODOLOGIA

Atal honetan, proiektuan egin den simulatzailearen azalpena, simulatuko den polimerizazio-prozesua nolakoa den eta inplementatutako algoritmoenak ikusiko dira.

3.1 SIMULAZIOA

Proiektuan konputazio ahalmena mugatua denez eta simulatzaile erreal bat konplexuegia denez, konputazio kostu baxuko simulatzaile sinple bat erabiliko da. Aukerak urriak dira, beraz, proiektura egokitzen den *NetLogo* [Wilensky, 1999] simulatzailearen aldaera bat sortzea erabaki da. Simulatzailea sinplea izan arren, simulatzaile konplexu edo sinple batekin erreakzio-baldintzen optimizazioa egitean dauden antzekoak problemak egongo dira. Beraz, proiektuan egingo den analisiak eta identifikatuko diren ondorio eta arazoak antzeko problemen ebazpenerako lagungarriak izatea espero da.

NetLogo agenteetan oinarrituriko programazio lengoaia da, eta eredu mota ezberdinak integratzeko eta simulatzeko aukera ematen du. NetLogo-k eredu batzuk ditu integratuta bere lan ingurunean eta polietileno tereftalatoa (PET) lortzeko, erradikal askeen bitarteko polimerizazio eredua, *RadicalPolimerization* [Wilensky, 1998], da hauetako bat.

RadicalPolimerization eredua ez da guztiz egokitzen proiektuaren beharretara; sinpleegia da, parametro gutxi baititu. Horregatik, simulatzailea *C* lengoaian zerotik berrinplementatu da, parametro berriak gehituz.

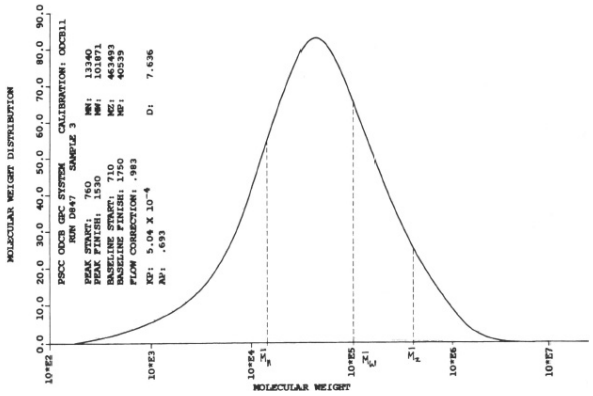
3.1.1 Erradikal Askeen Polimerizazioa

Erreakzio-ontzi batean erradikal askeen erreaktibitatearen ondorioz lortzen den molekula txikien (monomeroen) adizioari erradikal askeen polimerizazioa deritzo [Gooch, 2011]. Jarraian, etilenoaren bitartez PET lortzeko prozesu fisikoa azalduko da, hori baita simulatuko den polimerizazio-prozesua.

Polimerizazioak hiru urrats ditu: hastea, hedatzea eta amaitzea. Pauso horiek ez dira bata bestearen ondoren gertatuko, aldi berean baizik. Izan ere, erreakzio-ontzian, hasten ari diren kateekin batera, hedatzen ari diren beste batzuk egongo dira eta amaituta dauden beste batzuk.

1. $I_2 \rightarrow I^{\bullet} + I^{\bullet}$
2. $I^{\bullet} + M \rightarrow IM^{\bullet}$
3. $IMMM^{\bullet} + M \rightarrow IMMMM^{\bullet}$
4. $IMMM^{\bullet} + IMM^{\bullet} \rightarrow IIMMMMM$

3.1. irudia: Partikulen arteko erreakzioak.



3.2. irudia: PET polimeroaren pisu molekularren distribuzioa

Hastea

Hasarazle bat, oxigenoa adibidez, kanpoko eragileen eraginez, hautsi eta bi erradikal asketan bihurtzen da, 3.1 irudiko 1. erreakzioan erakusten den bezala. Erradikal aske hori etileno monomero batera transferituko da, monomeroa aktibatuz (2. erreakzioa, 3.1 irudian).

Hedaketa

Erradikal askea monomerora transferitu ondoren, monomeroa erradikal bihurtzen da erradikal aske kate bati hasiera emanez. Polimerizazio-prozesu osoan zehar erradikal aske kate honek beste etileno monomeroekin erreakzionatuko du. Erreakzio horrekin katearen luzera unitate batean handituko da (ikusi 3. erreakzioa, 3.1 irudian).

Amaitzea

Bi erradikal kateen arteko erreakzioa da. Erreakzioan, bi kateak elkartu eta egonkortzen dira, polimero-kate batean bihurtuz. Kate hori PET molekula bat izango da eta bi erradikal kateen baturaren luzera izango du (ikus 3.1 irudiko 4. erreakzioa).

Polimerizazio-prozesua amaitzean, luzera ezberdinetako polimeroen distribuzio bat lortzen da. Distribuzio horri, pisu molekularren distribuzioa deritzo.3.2 irudiak, horrelako distribuzioaren adibide bat erakusten du.

Algorithm 1 Polimerizazioa

```
1: procedure POLIMERIZAZIOA(M, R, P, T, MinMonRad, denbMax)
2:     amaitu = 0
3:     while amaitu = 0 do
4:         PartikulakMugitu()
5:         Hastea()
6:         Hedaketa()
7:         Amaitzea()
8:         if MinMonRad > KopMonRad OR denbMax < denb then
9:             amaitu = 1
       return (polimeroBekt)
```

3.2 SIMULATZAILEAREN INPLEMENTAZIOA

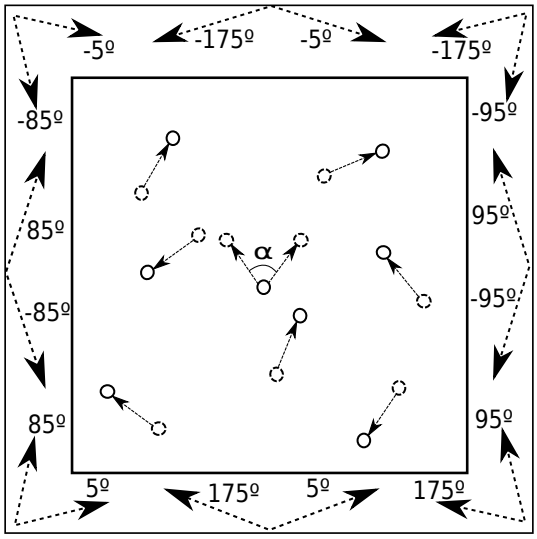
Aurreko atalean azaldu bezala, NetLogo simulatzailea baino parametro gehiago dituen simulatzaile bat inplementatu da, C lengoaian. Hona hemen, eginiko aldaketa nagusiak:

- Simulatzailearen parametro kopurua handitu da (2tik 4ra).
- Partikulen mugimendua errealagoa egin da.
- Simulatzailearen exekuzio-denbora murriztu da, kodea optimizatuz.

3.2.1 Simulatzailearen Algoritmoa

Algoritmoa inplementatzeko aurretik azaldutako polimerizazio-prozesuak jarraitzen dituen pausoak simulatu dira. Algoritmoak iterazioen bidez funtzionatzen du eta iterazio bakoitzean partikulak mugitu eta beraien arteko erreakzioak izango dituzte, hastea, hedaketa eta amaitzea hain zuzen. Mugimenduak ausaz egingo dira eta erreakzioak hurbil dauden partikulen artean gertatuko dira probabilitate jakin batekin.

Algoritmoak sarrera hauek ditu: M monomero kopurua, R monomero/hasarazle ratioa, P presioa, T tenperatura, MinMonRad erreakzionatzeko geratzen diren monomero eta erradikal kopuru minimoa eta denbMax exekuzio denbora maximoa. Lehenengo lauak dira optimizatu beharko direnak; azken biak simulazioa gelditzeko baldintza bezala erabiltzen dira. Polimerizazio-prozesua erreakzio-ontzi batean gertatzen da eta hori simulatzeko bi dimentsioko ontzi bat erabili da 30x30 tamainakoa NetLogo-ren ereduak erabiltzen duena mantenduz. Simulatzailearen irteera, amaitze-prozesuan lortutako polimeroen tamainen bektore bat izango dira eta hortik, pisu molekularren distribuzioa lortuko da.



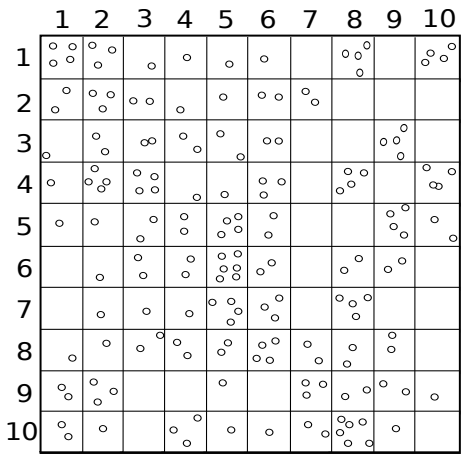
3.3. **irudia**: Partikulen mugimendu eta norabide posibleak.

3.2.2 Partikulen Mugimendua

Partikulak erreakzio-ontziaren barruan mugituko dira eta partikula bakoitzak uneoro kokapen eta norabide bat izango du. Iterazio bakoitzean, partikulak independenteki mugituko dira. Hona hemen partikula mugimenduen prozesua.

1. Simulazioaren hasieran, partikula bakoitzari $[0, 30]$ tartean ausazko x eta y koordenatuak eta $[-180^o, 180^o]$ tarteko ausazko α norabidea atzituko zaio.
2. j. iterazioan, i. partikula unitate bat mugituko da esleituta duen α norabidean.
$$x_i^{j+1} = x_i^j + cos(\alpha_i^j)$$
$$y_i^{j+1} = y_i^j + sin(\alpha_i^j)$$
3. Hurrengo iterazioan, i. partikularen norabidea eguneratuko da, partikularen norabidea $[-50^o, 50^o]$ tarteko ausazko α_{rand} balioaz desbideratuta.
$$\alpha_i^{j+1} = \alpha_i^j + \alpha_{rand}$$

Partikulak ontzi batean daudenez, alboetara iristean ontziaren aurka jo eta errebotatu egiten dute. Errebote hori ausazko angelu batekin egingo da, partikulak oso txikiak direlako eta ontziaren hormak irregularrak izaten direlako eskala molekularrean. Beraz, horma batera iristean partikulek 3.3 irudian ikus daitezkeen angeluen arteko ausazko balioak artuko dituzte.



3.4. irudia: Partikulak erreakzio-ontzian. Ontzia $10x10$ txataletan banatuta.

3.2.3 Partikulen Arteko Erreakzioak

Simulatzaileak lau partikula mota ditu: monomeroak, hasarazleak, erradikal kateak eta polimero-kateak. Lehen azaldu bezala, hiru erreakzio mota gerta daitezke beraien artean: hastea, hedaketa eta amaitzea.

Partikulen arteko erreakzioak hurbiltasun bitartez egingo dira. Bi partikula hurbil dauden edo ez jakiteko ontzia txataletan (patch) banatu da (ikus 3.4 irudia). Txatal bakoitzak hainbat partikula jasoko ditu eta txatal berean dauden partikulak beraien artean erreakzionatzeko aukera izango dute.

Txatal berean dauden partikulak beraien artean erreakzionatu dezaketen arren, ez dute beti erreakzionatuko, erreakzioak probabilitate batekin gertatuko baitira. Probabilitate hori erreakzio mota bakoitzerako ezberdina da eta tenperatura eta presioaren araberakoa izango da; probabilitate horri erreakzio-konstantea deituko zaio eta beti 0 eta 100 balioen artekoa izango da, NetLogo ereduak egiten duen bezala. Tenperatura eta presioa simulatzaile berrira gehitutakoak direnez, NetLogo eredua imitatzeko bi parametro horientzat $[30, 100]$ eta $[1, 10]$ mugak jarri dira, konstanteen ekuazioak 0 eta 100 tartean egoteko.

Hastea

Erreakzioa hasarazle bat bi erradikal asketan banatzean datza. Erreakzioa amaitzean hasarazlea desagertu egiten da eta sortu berri diren bi erradikalek unitate bateko tamaina izango dute. Jarraian, erreakzioaren eskema zehazten da:

1. Erreakzio-ontzian dagoen hasarazle bat hartu.
2. Ausaz $[0, 1000]$ arteko zenbaki bat aukeratu.
3. Zenbaki hori $K_i = 0.01 * T^2 + P - 10$ erreakzio-konstantearen balioa baino txikiagoa bada, hasarazlea bi erradikal asketan banatu.

Hedaketa

Erreakzio honetan erradikal batek eta monomero batek parte hartzen dute. Iterazio jakin batean erradikal bat eta monomero bat txatal berean daudenean gertatzen da.

Erreakzioan, erradikalak bere tamaina unitate bat handituko du monomeroa erradikal katera eransten delako. Erradikal honek ezin izango du iterazio berean beste erreakziorik izan. Jarraian, erreakzioaren eskema zehazten da:

1. Txatal bakoitzetik erreakzionatu ez duen erradikal kate bat eta monomero bat aukeratu.
2. Ausaz $[0, 1000]$ arteko zenbaki bat aukeratu.
3. Zenbaki hori $K_m = T + \frac{10}{33} * P^2 - \frac{1000}{33}$ erreakzio-konstantearen balioa baino txikiagoa bada, monomeroa erradikal askeen katera erantsi.

Amaitzea

Azken erreakzio hau bi erradikalen artekoa da. Bi erradikal hurbil daudenean, hauek elkartu eta polimero-kate bihurtzen dira. Polimero-kate honek bi erradikalen tamainen batura izango du eta ez du gehiago erreakzionatuko, hau da, bere tamaina mantenduko du simulazioa bukatu arte. Jarraian, erreakzioaren eskema zehazten da:

1. Txatal bakoitzetik erreakzionatu ez duen bi erradikal kate aukeratu.
2. Ausaz $[0, 500]$ arteko zenbaki bat aukeratu.
3. Zenbaki hori $K_m = log(T) + 0.09895 * P^3$ erreakzio-konstantearen balioa baino txikiagoa bada, bi erradikal kateak erantsi eta polimero-kate bihurtu.

$T = 40$ eta $P = 5$ finkatuta, prozesuaren adibide bat azalduko da jarraian, egiten dena hobeto ulertzeko:

- Hastea: Ausaz 0 eta 1000 artean ateratako zenbakia 15 bada, $0.01 * 40^2 + 5 - 10 = 11$ lortzen dugu, $11 < 17.27$ denez, hasarazlea bi erradikal asketan banatuko da.
- Hedaketa: Ausaz 0 eta 1000 artean ateratako zenbakia 30 bada, $(40 + \frac{10}{33} * 5^2 - \frac{1000}{33}) = 17,27$ lortzen dugu, $30 > 17.27$ denez erreakzioa ez da gertatuko.
- Amaitzea: Ausaz 0 eta 500 artean ateratako zenbakia 10 bada, $log(40) + 0.09895 * 5^3 = 16.05763$ lortzen dugu, $10 < 16.05763$ denez bi kateak elkartu eta polimero-kate bihurtuko dira.

3.3 PROBLEMAREN FORMALIZAZIOA

Optimizazio problema bat ebazteko, problemaren formalizazioa burutuko da eta problema hori formalizatzeko, soluzioen kodeketa eta helburu-funtzioa definituko dira. Lehenengo, soluzioaren kodeketa azalduko da eta ondoren helburu-funtzioa.

3.3.1 Soluzioaren Kodeketa

Optimizazio problemetako lehenengo pausoa soluzioaren kodeketa zein izango den definitzea da. Simulatzaileak lau parametro ditu optimizatzeko, beraz, hautaturiko kodeketa lau parametroen balioen bektorea izango da, $s = (M, R, P, T)$, M monomero kopurua (balio osoa), R ratioa (balio jarraitua), P presioa (balio jarraitua) eta T tenperatura (balio jarraitua) izanik.

Balio horiek mugatuta egongo dira, lehen esan bezala, eta, beraz, presioak eta tenperaturak $[1, 10]$ eta $[30, 100]$ tarteko balioak eduki behar dituzte. Ratioak ere $(0, 1]$ tarteko balioak hartu behar ditu, hasarazle kopuruak monomeroen zein portzentaia den esan nahi duelako. Monomero kopurua $[500, 20000]$ balioen artean finkatuko da, 500 monomero baino gutxiagorekin ontziko monomero kontzentrazioa oso baxua baita, eta, ondorioz prozesua asko luzatuz; 20000rekin ordea kontzentrazioa altuegia eta zentzurik gabeko emaitzak lortzen dira.

3.3.2 Soluzioaren Ebaluazioa

Problemaren ebazpenean, helburu-funtzioa behar da, simulatzaileari bilaketa espazioko soluzio bat emanda itzulitako emaitzaren kalitatea jakiteko. Simulatzaileak itzuliko duen emaitza polimero luzeren distribuzioa (pisu molekularren distribuzio) bada, helburu bezala lortu nahi den pisu molekularren distribuzioarekin konpara daiteke. Horretarako, distribuzioak konparatzeko oso erabilia den *Kullback-Leibler (KL)* dibergentzia [Kullback and Leibler, 1951], D_{KL} ere deitua, aukeratu da helburu-funtzio bezala. Kullback-Leibler dibergentziaren bitartez, bi probabilitate distribuzio konparatuko dira, lortu nahi den pisu molekularren distribuzioa eta soluzio batetik lorturiko distribuzioa.

Konparatu nahi diren distribuzioak multinomialak dira. Kasu horretan $\mathbf{p} = (p_1, ..., p_n)$ eta $\mathbf{q} = (q_1, ..., q_n)$, distribuzioak izanik, D_{KL} modu honetan kalkulatuko da:

$$D_{KL}(\mathbf{p}||\mathbf{q}) = \sum_{i=1}^{n} p_i log\frac{p_i}{q_i} \text{ , } q_i > 0$$

Pisu molekularren distribuzioa lortzeko polimero-kate luzera bakoitzeko zenbat polimero dauden zenbatu behar dira. Adibidez, $(1, 2, 3, 3, 1, 5)$ tamaina-bektorea izanik, pisu molekularren distribuzioa honako izango da: $(\frac{2}{6}, \frac{1}{6}, \frac{2}{6}, \frac{0}{6}, \frac{1}{6})$. Hau da, sei polimero-katetatik 1 luzerako bi polimero-kate daude bektorean, beraz, bektoreko 1. posizioan

2 zenbakia jarriko da eta distribuzioaren 4. posizioak 0 balioa izango du ez baitago 4 luzerako katerik.

Pisu molekularren distribuzioa lortzean, **q** distribuzioak zero balioak izan ditzake eta bi distribuzioen luzerak ezberdinak izan daitezke. Ondorengo pausoak jarraituz egokituko dira distribuzioak, konparagarriak izateko:

1. Distribuzioak tamaina bera eduki dezaten moztu egingo dira **p** eta **q**. Mozketa egiterakoan probabilitate distribuzioko lehenengo 100 balioak hartuko dira [I].
2. Zero probabilitateen arazoa ekiditeko, Laplace-ren zuzenketa [Zabell, 1989] aplikatuko zaio distribuzioari, kate tamaina kopuru guztiei gehi bat eginez.

Soluzioak ebaluatzerakoan, **p** probabilitate distribuzioa erreferentziazko distribuzioa izango da eta **q**, berriz, simulatzailearekin lortu den soluzio berria. KL dibergentzia 0ra zenbat eta gehiago hurbildu, orduan eta antz handiagoa izango dute **p** eta **q** distribuzioek. Hortaz, helburua D_{KL} minimizatzea izango da.

3.4 OPTIMIZAZIO ALGORITMOAK

Atal honetan problema ebazteko aukeratu diren hiru algoritmoak azalduko dira. *Ausazko Bilaketa*, *Line Search* eta *Algoritmo Genetikoa*, hain zuzen.

3.4.1 Ausazko bilaketa

Optimizazio algoritmo sinpleenetako bat da ausazko bilaketa. Algoritmo honek ausazko parametro konbinaketak sortuko ditu eta parametro horiekin simulatzailea exekutatuko da. Ondoren, lortu nahi den polimeroaren instantziarekin konparatuko da soluzioa, KL dibergentzia erabiliz.

Algoritmo hau oinarri moduan erabiliko da. Hau da, beste algoritmoak hau baino hobeak izatea espero da.

3.4.2 Line Search

Ausazko soluzio bideragarri batetik hasten da soluzio on baten bila *Line Search* [Snyman, 2005] optimizazio algoritmoa. Algoritmo honek hasieran ausaz lortuko den $s = (s_1, ..., s_n)$ soluzioko s_i parametro bakoitzaren inguruko d distantziara dauden inguruko soluzioak aztertuko ditu iteratiboki. d distantzia definitzeko metodo ezberdinak daude: Powell-en interpolazio metodoa, eskuz definitzen dena, gradientean oinarriturikoa,

I Kate gehienak 100 luzera baino txikiagoak dira, beraz informazio galera oso txikia da

golden search etab. Proiektua inplementaturiko Line Search algoritmoan, eskuz finkatu da d distantzia.

Algoritmoaren j. iterazioko soluzioa $s^j = (s_1^j, ..., s_n^j)$ bada aztertuko diren soluzioak i. parametrorako

$s^j + \Delta_i$ eta $s^j - \Delta_i$ izango dira, non $\Delta_i = (d_1, ..., d_n)$ den eta

$d_k = 0$, $i \neq k$ eta $d_i = d$ izanik,

i parametroaren bi inguruko soluzioen helburu-funtzioak uneko soluzioarekin alderatuko dira, bietako bat hobea dela egiaztatuz:

$f(s^j) > f(s^j + \Delta_i)$ edo $f(s^j) > f(s^j - \Delta_i)$

Baldintza betetzen bada, f gehien hobetzen duena aukeratzen da hurrengo iteraziorako eta noranzkoa gordeko da, hau da, $s^{j+1} = s^j - \Delta_i$ bada, aurrerantzean hurrengo soluzioari Δ_i kenduz. Biak txarragoak badira, soluzioaren hurrengo parametrora pasatzen da.

Azken parametroarekin bukatu ondoren algoritmoa lehenengo parametrotik hasiko da berriro azterketa egiten, amaiera baldintzak betetzen ez diren arte.

Line Search algoritmo lokala da, ondoz ondoko beste soluzio baten bila ibiltzen baita soluzioan hobekuntzarik dagoen ikusiz. Ondoz ondoko soluzioak soilik aztertzen dituelako optimo lokaletan trabaturik gelditzen da. Parametroak independenteki aztertzen dira bata bestearekin dituzten erlazioak kontuan eduki gabe. Bilaketa mota horiek problema konbexu eta jarraituetan dira onak.

Proiektu honetan diseinatu den Line Search algoritmoa, 2 algoritmoan erakusten da.

3.4.3 Algoritmo Genetikoa

Algoritmo genetikoak [Sivanandam and Deepa, 2008], algoritmo ebolutiboen taldekoak dira eta aldi berean, algoritmo ebolutiboak populazioan oinarrituriko algoritmoak dira. Atal honetan algoritmo genetikoen ezaugarriak azalduko dira eta azkenik, inplementatu den algoritmoaren kodea azalduko da.

Populazioan oinarritutako algoritmoek iterazio bakoitzean soluzio baten inguruko soluzioak bilatu ordez, soluzio multzoak maneiatzen dituzte. Soluzio multzo horiei *populazio* izena ematen zaie.

Algoritmo ebolutiboek, aurreko prozesuaz gain, iterazio bakoitzean soluzioak ebolu-zionarazten dituzte soluzio hobeak lortzeko asmoz eta teknika ezberdinak erabilita.

Algoritmo genetikoak Darwinen eboluzio teoria imitatuko du, eta hasierako populazioa belaunaldiz belaunaldi (iterazio bakoitzean), aldatuz joango da teknika ezberdinen bidez: *mutazioz*, bi soluzio gurutzatuz (*crossover*), etab. Soluzio berriek bere aurreko belaunaldiko gurasoen ezaugarri onak mantentzea eta hobetzea da algoritmo genetikoen xedea, eta gaizki egokitzen diren soluzioak kanpoan uztea.

Algorithm 2 Line Search

```
1: procedure LineSearch(instantzia, denbMax, ebalMax, minMonRad, d[M, R, P, T])
2:     evalKop = 1
3:     s = ausaz[M, R, P, T]                          ▷ Ausazko soluzio bat sortu
4:     minSol = f(s)          ▷ Ausazko soluzioa ebaluatu helburu-funtzioarekin
5:     i = 1 , j = 1
6:     Δ = (0, 0, 0, 0)
7:     while ez Amaitu do
8:         Δ[i] = d
9:         if j == 1 then                  ▷ Lehen iterazioan bi noranzkoak ebaluatu
10:            if then f(s + Δ) > f(s) ∧ f(s − Δ) > f(s)   ▷ Ez bada hobetzen soluzioa
11:                Δ[i] = 0
12:                i = i + 1                              ▷ Hurrengo parametrora igaro
13:                Δ[i] = d
14:            else                          ▷ Hobetzen bada noranzko horretan jarraitu
15:                if f(s + Δ) < f(s − Δ) then
16:                    α = +1
17:                else
18:                    α = −1
19:            evalKop = evalKop + 2
20:        else                                  ▷ α Noranzkoan ebaluatzen jarraitu
21:            s = f(s + α ∗ Δ)
22:            j = j + 1
23:            evalKop = evalKop + 1
24:        if minSol > s then
25:            minSol = s
26:        else                                          ▷ Ez bada hobetzen emaitza
27:            i = i + 1                                 ▷ Hurrengo parametroa aztertu
28:            j = 1
29:            if i == 4 then
30:                i = 1
31:        evalKop = evalKop + 1
       return (minSol , evalKop)
```

Hori guztia lortzeko, pauso hauek jarraitu behar ditu algoritmo genetiko batek:

1. Hasierako populazioa aukeratu.
2. Soluzio berriak sortu.
3. Populazio berria aukeratu soluzio zahar eta berrietatik.
4. Amaitze baldintzak betetzean, algoritmoaren exekuzioa amaitu. Bestela, 2 eta 3 pausoak errepikatu.

Hasierako populazioa aukeratu

Populazioan oinarritutako algoritmoetan soluzioen dibertsitateak garrantzi handia du, eta baita kalitateak ere. Hori kontuan hartuta, teknika sinpleena ausazko soluzioak eskuratzea da, onak ez izan arren, oso soluzio ezberdinak eskuratzen baitira normalean. Beste teknika bat pseudoausazko soluzioak sortzea da. Horretarako, soluzio batetik bestera distantzia minimo bat ezarrita, dibertsitate handiagoa lortzen da. Kalitatea hobetzeko, hasieran ausaz sortutako soluzioei hobekuntza algoritmo bat aplikatzen zaie, dibertsitate ezberdinetako eta kalitatezko soluzioak lortuz.

Aurkeztuko den algoritmoan ausazko soluzioak sortuko dira eta populazio tamainak dibertsitatean eragiten duenez bi populazio tamaina ezberdinekin egingo da analisia, parametro honen eragina ikusi ahal izateko.

Soluzio berriak sortu

Algoritmo genetikoetan, populazioan dauden hainbat soluzio aukeratzen dira (normalean bi) eta algoritmoan aukeraketa egiteko *lehiaketa hautespena* erabiliko da. Lehiaketa hautespenak bitan populazioko n indibiduo aukeratuko ditu ausaz eta horietatik onena aukeratuko du eta lorturiko bi soluzioen ezaugarriak dituen soluzio berri bat sortuko da. Soluzio berriak sortzeko modu horri crossover (gurutzatzea) deitzen zaio. Crossover teknikaren bitartez, populazioko bi soluzio edo gehiago aukeratu eta horiek dituzten ezaugarriak transmititzen zaizkio soluzio berriari.

Gurutzaketa teknika ezberdinak daude eta proiektu honetan erabiliko duguna bataz besteko gurutzaketa aritmetikoa izango da. Bataz besteko gurutzaketaren bitartez, bi gurasoen ezaugarri guztien arteko bataz bestekoak hartuko ditu soluzio berriak:

$s' = \frac{s_1+s_2}{2}$

Ondoren, eboluzioan gertatzen den moduan, sorturiko soluzio berri horiek probabilitate batekin mutazioak jasango dituzte eta dibertsitatea mantenduko da.

Sortuko diren soluzio berrien kopuruari muga bat ezarri behar zaio algoritmo hauetan. Proiektuan aurkeztuko den algoritmoan, sortuko diren soluzio berrien kopuruak populazio tamainaren erdia izango da.

Populazio berriaren aukeraketa

Soluzio berriak sortu ondoren, hurrengo belaunaldiko populazioa zein izango den finkatu behar da. Teknika erabiliena aukeraketa elitista da, hain zuzen ere aurkeztutako algoritmoan erabiliko dena. Bertan, aurreko belaunalditik eta sortu berri diren soluzioetatik onenak aukeratzen dira hurrengo belaunaldirako.

Algorithm 3 Algoritmo Genetikoa

```
1:  procedure AlgoritmoGenetikoa(denbMax, ebalMax, minMonRad, popKop, mutazioProb)
2:      ebaluazioKop = popKop
3:      pop = sortu_populazioa(popKop)
4:      popEval = eval(pop)
5:      while ez Amaitu do
6:          for 1 : (popKop/2) do                          ▷ Soluzio berriak sortu
7:              (s1, s2) = lehiaketa_hautespena(pop)
8:              s' = gurutzaketa(s1, s2)
9:              mutatu(s', mutazioProb)
10:             pop = populazioan_sartu(s')
11:             ebaluazioKop = ebaluazioKop + 1
12:         aukeraketa_elitista(pop)
13:         popEval = eval(pop)
14:     minKL = minKL(pop)
15:     minSol = minS(pop)
        return (minsol , minKL)
```

Algoritmoaren exekuzioa amaitu

Algoritmoaren exekuzioa amaigabea denez, irizpide batzuk ezartzen dira algoritmoa amaitzeko. Irizpide horiek denbora, ebaluazio kopurua, konbergentzia eta abar dira. Denbora maximoa eta ebaluazio kopurua izango dira algoritmoan erabiliko diren irizpideak exekuzioa amaitzeko. Proiektuan inplementatu den algoritmo genetikoa 3 algoritmoan azaltzen da.

4 ESPERIMENTAZIOA

Kapitulu honetan simulatzailearen bilaketa espazioa eta optimizazio algoritmoak aztertuko dira, bakoitzaren ezaugarri onak eta egongo diren arazoak identifikatzeko asmoz. Azterketa hori egiteko, proba ezberdinak egin dira, hauek azaldu eta analisiak grafikoetan adierazi dira lortutako emaitzak errazago uler daitezen. Kapituluak hiru atal ditu: instantzien sortzea, bilaketa espazioaren azterketa eta algoritmoen azterketa.

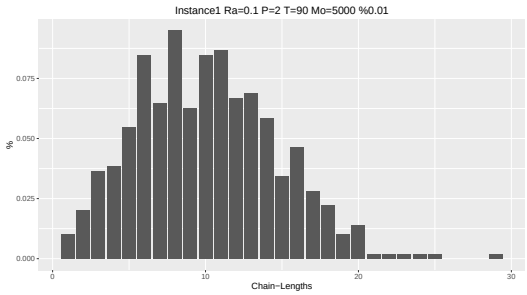
4.1 INSTANTZIAK SORTZEA

Probak egiten hasi aurretik, helburu bezala lortu nahi diren pisu molekularren distribuzio batzuk behar dira, hau da, ezaugarri jakin batzuetako polimeroak eduki behar dira simulatzailea eta algoritmoak probatzeko. Horretarako, proiektuan inplementatu den simulatzailea erabiliz, 10 instantzia sortu dira, beraien artean desberdinak direnak. Instantzia horiek sortzean, ez zaie amaiera parametrorik jarri eta polimerizazio-prozesua amaitu arte itxaron da, lortutako emaitzak ahal den zehatzenak izan daitezen.

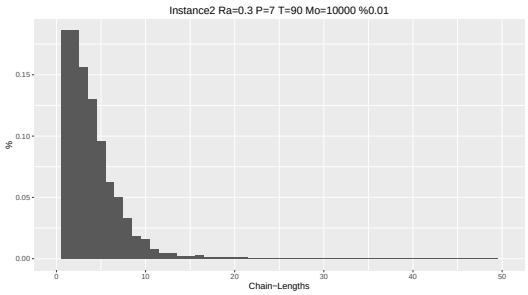
Jarraian, instantzia bakoitza sortzeko parametroak zeintzuk diren zehaztuko da 2 taulan eta, ondoren, parametro jakin horietatik sortu diren distribuzioak irudikatu dira 4.5 irudian.

Instantzia	Mo	Rat	P	T
1	5000	0.1	2	90
2	10000	0.3	7	90
3	2000	0.4	1	60
4	5000	0.05	10	80
5	7000	0.3	1	70
6	18000	0.1	9	100
7	1500	0.2	4	40
8	4000	0.15	5	95
9	19000	0.55	5	50
10	5000	0.25	2	75

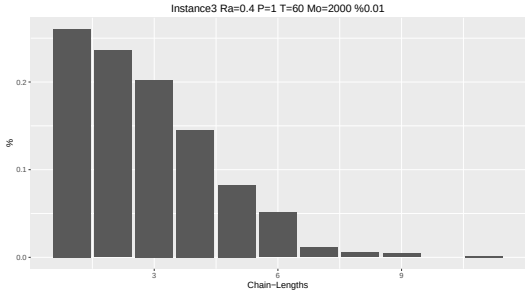
2. **taula**: Instantziak sortzeko erabilitako parametroak



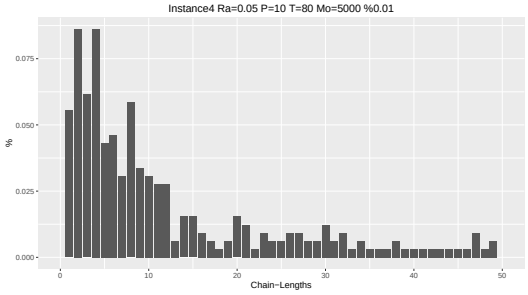
(a) 1. *Instantzia*



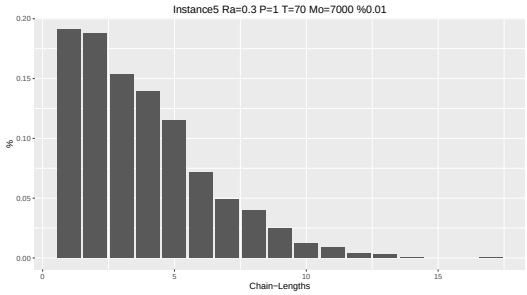
(b) 2. *Instantzia*



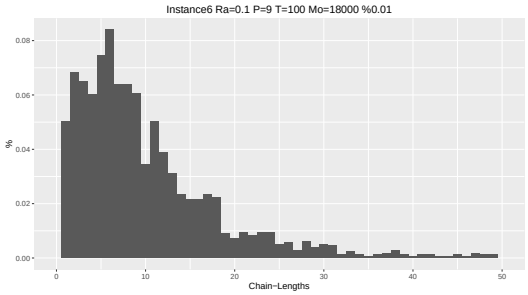
(c) 3. *Instantzia*



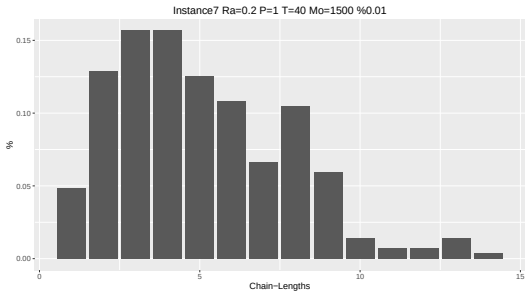
(d) 4. *Instantzia*



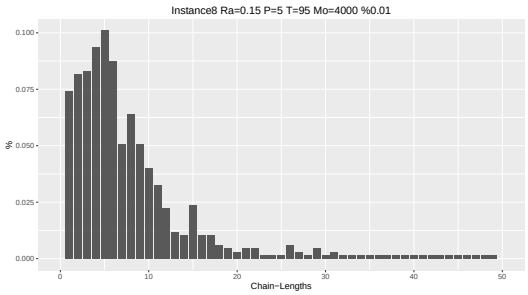
(e) 5. *Instantzia*



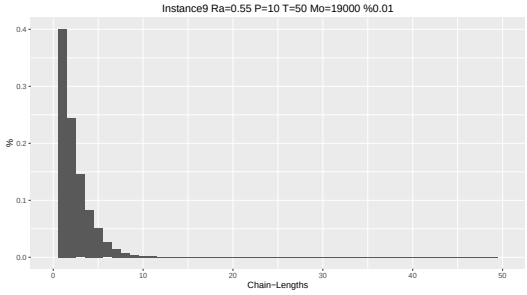
(f) 6. *Instantzia*



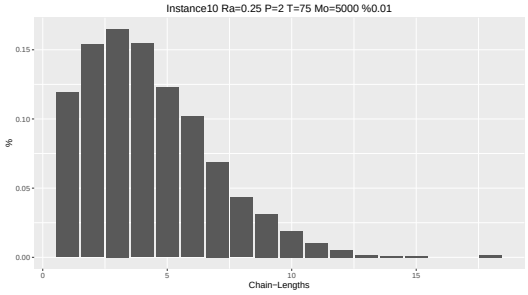
(g) 7. *Instantzia*



(h) 8. *Instantzia*



(i) 9. *Instantzia*



(j) 10. *Instantzia*

4.5. irudia: Instantzien Distribuzioak

4.2 BILAKETA ESPAZIOAREN AZTERKETA

Algoritmoek emango dituzten emaitzen analisi zuzen bat egiteko simulatzailearen portaera nolakoa den jakitea oso lagungarria da, edukitako arazo eta asmatutako aldeak identifikatzen lagunduko baitu. Horregatik, bilaketa espazioaren azterketa egin da, simulatzailean soluzio konbinazio ezberdinak simulatu eta emaitzetan duten eragina aztertuko da.

Proba guztiak 10 instantziekin egingo dira eta aldatutako parametro bakoitzeko 10 errepikapen. Denbora maximoa 40 segundotan ezarri da exekuzioetan efekturik izan ez dezan eta grafikoak argiago ikus daitezen $1, 2, 6$ eta 9 instantziak azalduko dira soilik, beste guztiak eranskinetan ikusteko aukera izanik. Erreakzio-ontzia lehen aipatu bezala $30x30$ekoa izango da.

4.2.1 Amaiera Parametroak

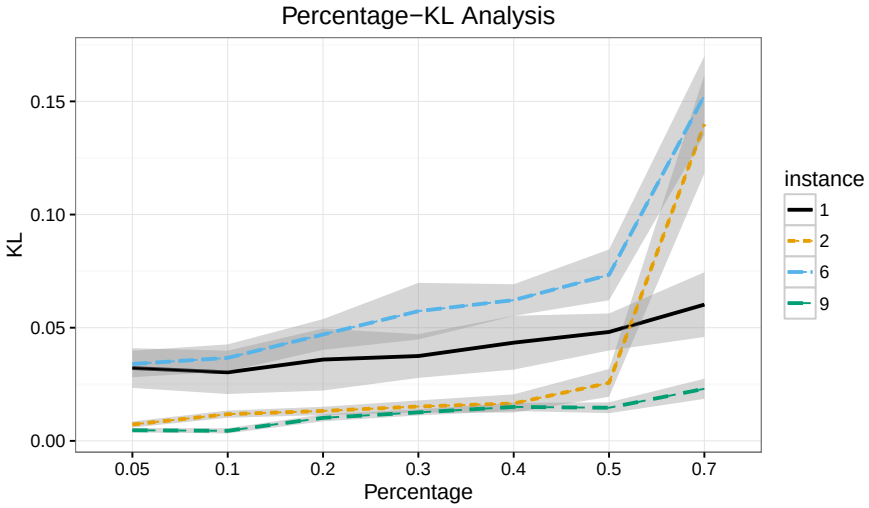
Polimerizazio simulazio bat erreakzio gehiago ezin egon arte utziko balitz, denbora tarte luze bat pasako luke polimerizazio-prozesuak exekuzioan eta hori saihesteko amaiera parametroak egokitu behar dira, denbora maximoa eta monomero eta erradikal portzentaia minimoen balioak hain zuzen.

Exekuzio-denbora murriztuko duen eta simulazioaren emaitza aldatuko ez duen balio bat bilatu nahi da. Horretarako, erradikal eta monomeroen portzentaia minimo ezberdinak probatuko dira instantzien aurka, denbora mugarik gabe, eta emaitzetan eta denboran duten efektua zenbatekoa den analizatuko da. Hauek dira probatu diren portzentaia minimoak: $0.05, 0.10, 0.20, 0.30, 0.40, 0.50, 0.70$.

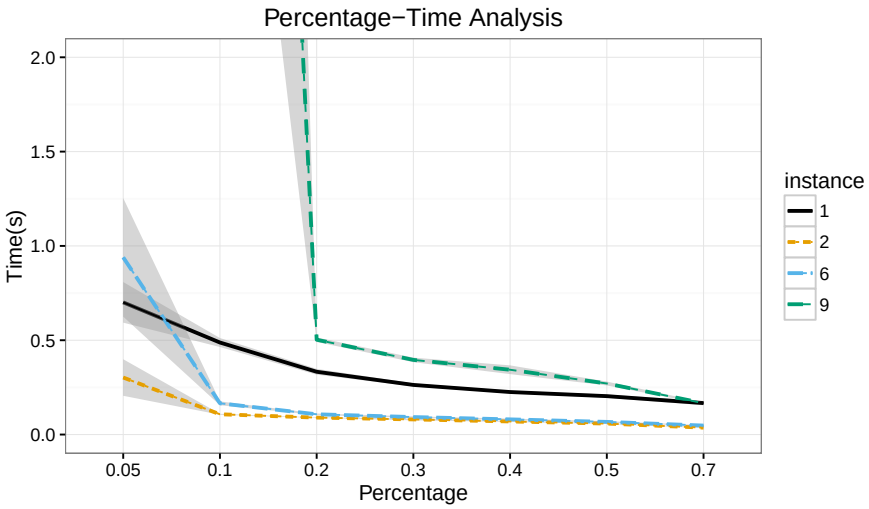
Lorturiko emaitzak 4.6 irudiko grafikoetan ikus daitezke. Lehenengo grafikoan portzentaia minimoek helburu-funtzioan duten efektua eta desbideraketa estandarra zein den ikusten da (tarte grisean). Desbideraketa simulatzailearen emaitzen aldakortasunaren adierazgarri bat bezala ikus daiteke. Halere, lehenengo grafiko honek KL dibergentzietan aldaketak txikiak direla adierazten du 0.4ko portzentaien azpitik.

Bigarren grafikoan, argiago ikus daiteke portzentaien eragina denborarekiko, portzentaia minimoak baxuak direnean denbora gehiago behar baitu simulatzaileak exekuzioa amaitzeko.

Bi grafiko hauek aztertu ondoren eta ondorengo probetarako beharrezkoa izango denez balio hau finkatzea, 0.20ko portzentaia minimoa finkatu da, exekuzio-denbora gutxitzen baitu eta emaitzetan duen eragina oso txikia baita.



(a) *KL dibergentzia*



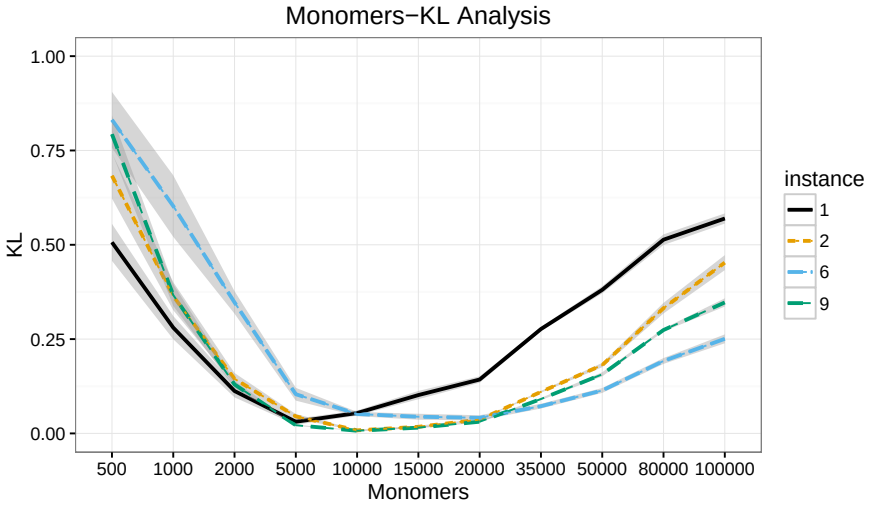
(b) *Exekuzio − Denborak*

4.6. irudia: Portzentaia minimoen analisia

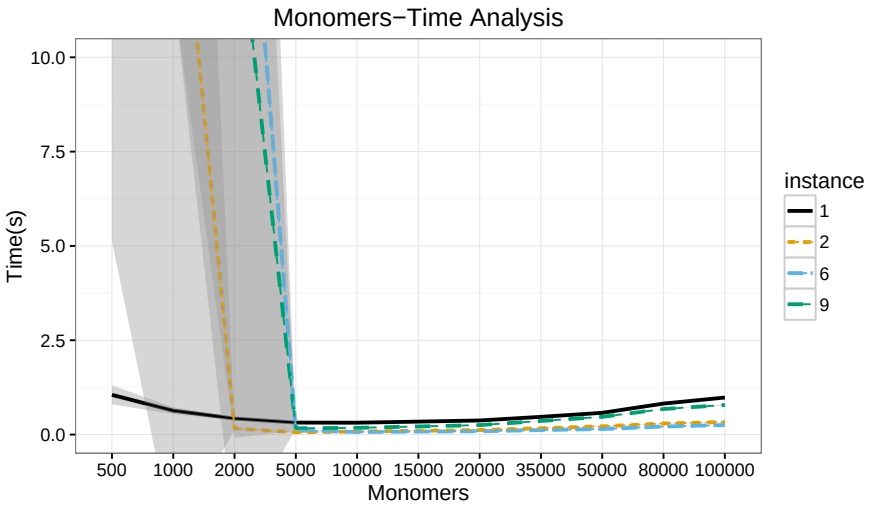
4.2.2 Monomero Analisia

Monomero kopuruak erreakzio-ontziko monomero-kontzentrazioa ezartzen du. Proba honetan, monomero-kontzentrazioak helburu-funtzioan duen eragina aztertu nahi da, optimizazioan izango duen pisua zenbatekoa den jakitearren. Horretarako, monomero kantitate ezberdinen analisia egin da instantzien aurka monomero kopuru hauekin: 500, 1000, 2000, 5000, 10000, 15000, 20000, 35000, 50000, 80000, 100000.

4.7 irudiko (a) grafikoan monomero kopuruak KL dibergentzian duen eragina ikus daiteke. 1 instantziak 5000 monomerorekin lortzen ditu emaitza honenak (5000 mo-



(a) *KL dibergentzia*

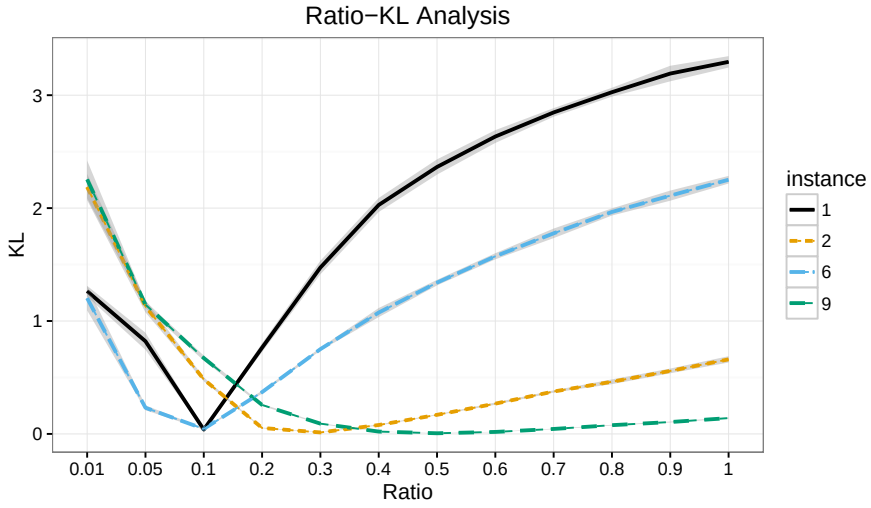


(b) *Exekuzio − Denborak*

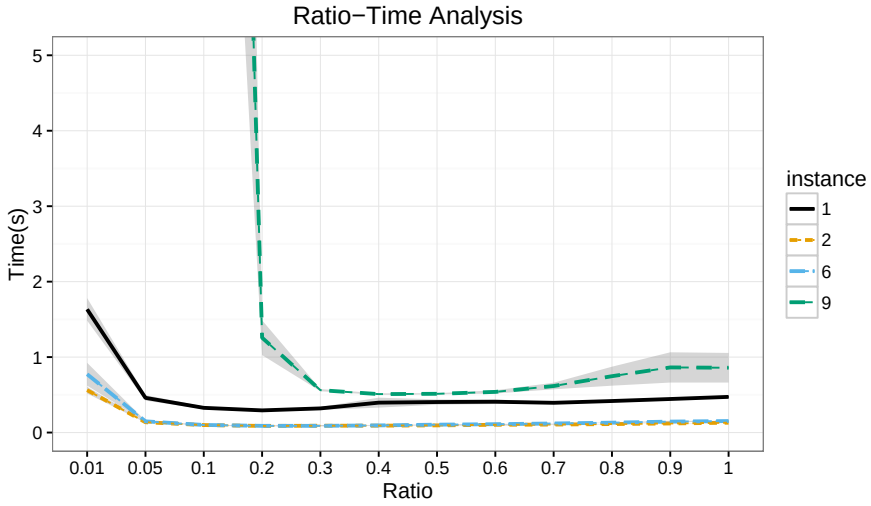
4.7. irudia: Monomeroen Analisia

nomero baitzituen instantzia sortzerakoan) eta beste kopuruekin emaitza okerragoak lortzen dira. Beste instantziekin ere antzekoa gertatzen da.

Honekin batera, b) grafikoan ikusten den bezala, monomero kontzentrazioa baxua denean (500-2000 monomero) exekuzio-denbora altua da, ontziko partikula kontzentrazio txikia denez ez dutelako beraien artean hain erraz erreakzionatzen. Kontzentrazioa handitzen doan heinean ordea denborak txikiagotu egiten dira, eta oso altuak direnean denborak gora egiten du. Hori, portzentaia minimoak berdinak direnean, M bikoiztean erreakzioak ere bikoiztu egiten direlako gertatzen da.



(a) *KL dibergentzia*



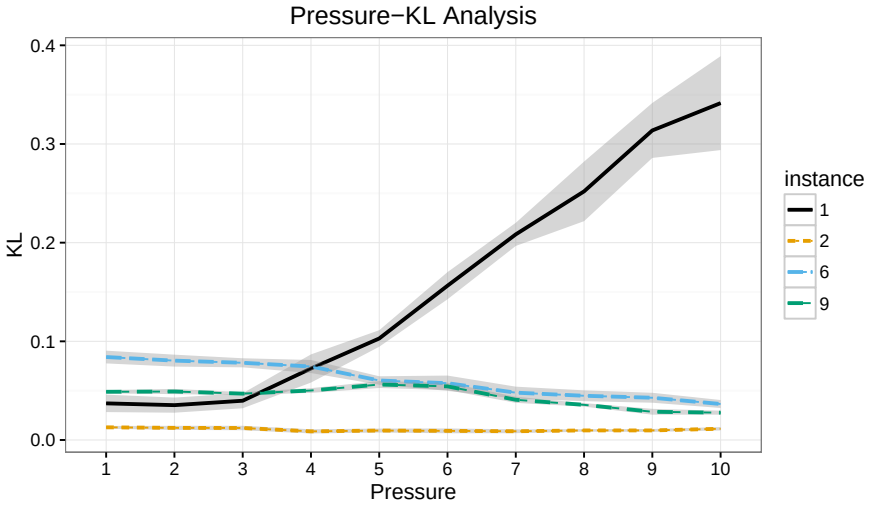
(b) *Exekuzio − Denborak*

4.8. irudia: Ratioaren Analisia

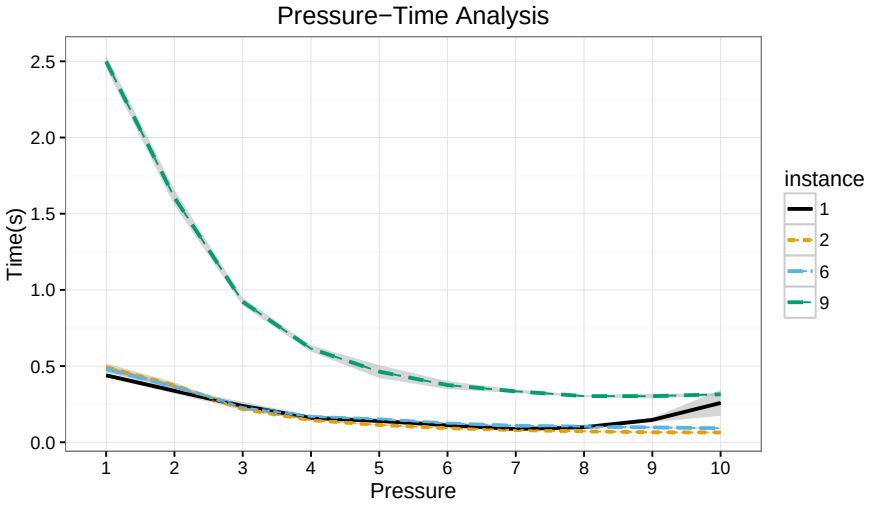
4.2.3 Hasarazleen Ratioaren Analisia

Ratioak monomeroekiko zenbateko hasarazle portzentaia dagoen ezarriko du. Hasarazle kontzentrazioak emaitzetan egiten dituen aldaketak ikusteko eta optimizazioan duen pisua zenbatekoa den jakiteko, ratio portzentaia ezberdinekin egin dira probak. Ratio portzentaia hauekin egingo dira probak: 0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5, 0.55, 0.6, 0.7, 0.8, 0.9, 1.

Lortutako emaitzak 4.8 irudiko grafikoetan azaltzen dira. Bi grafikoak aztertu ondoren, esan daiteke ratioak eragin handia duela lortuko diren emaitzetan eta, ondorioz, optimizazioan garrantzi handiena duen parametroa izango dela.



(a) *KL dibergentzia*



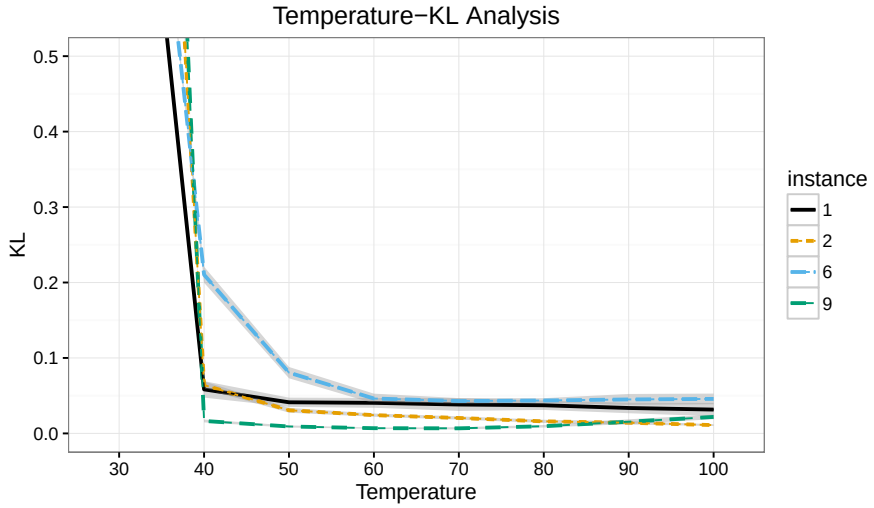
(b) *Exekuzio − Denborak*

4.9. irudia: Presioaren Analisia

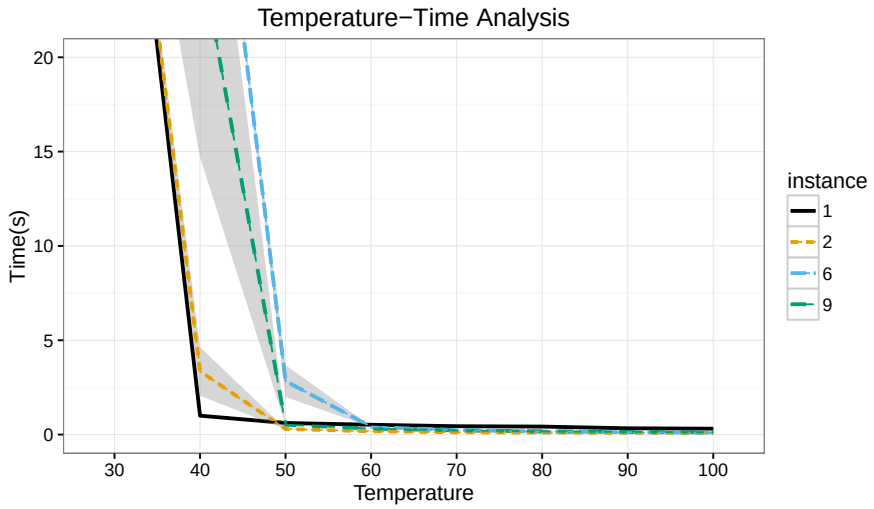
4.2.4 Presioaren Analisia

Parametro honek polimerizazio-prozesua zenbat atmosferatan gertatuko den finkatuko du. Helburu-funtzioan eta exekuzio-denboretan presioaren efektua analizatuko da, probak presioaren balio hauekin eginez: $1, 2, 3, 4, 5, 6, 7, 8, 9, 10$.

Lorturiko emaitzak, 4.9 irudiko grafikoan erakusten dira. Dibergentziaren analisian presioak emaitzak aldatzen dituela ikus daiteke, beraz garrantzia edukiko du optimizazioan. Denbora eta presioaren grafikoa ikusten badugu, presioak exekuzio-denboretan duen garrantzia jakitera eramango digu eta presioaren araberakoa izango da ezarriko den exekuzio-denbora maximoa.



(a) *KL dibergentzia*



(b) *Exekuzio − Denborak*

4.10. irudia: Tenperaturaren Analisia

4.2.5 Tenperaturaren Analisia

Parametro honek polimerizazio-prozesua tenperatura zenbat gradutan gertatuko den finkatuko du. Helburu-funtzioan eta exekuzio-denboretan tenperaturak duen efektua analizatuko da, simulazioak balio hauekin eginez: $40, 50, 60, 70, 80, 90, 100$.

4.10 irudiko grafikoetan ditugu lortutako emaitzak. Lehenengo grafikoak, tenperaturak emaitzetan inongo aldaketarik ez duela egiten esaten du. Bigarren grafikoak ordea, tenperaturak denboretan aldaketa handiak egiten dituela ikusten da.

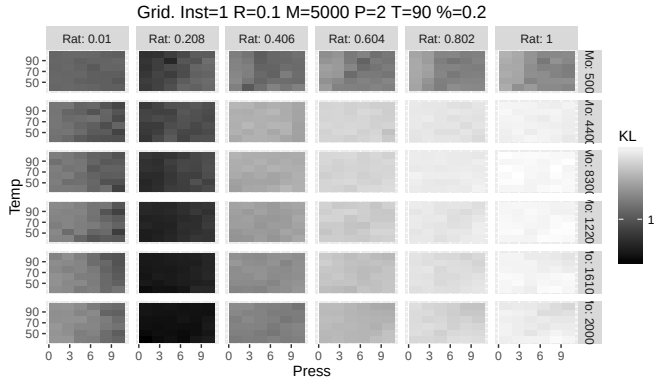
4.2.6 Instantzien Soluzio Espazioaren Analisia

Bilaketa espazioaren analisiarekin amaitzeko, lau instantzien bilaketa espazioa modu grafikoan bistaratuko da, monomero kantitate, ratio, tenperatura eta presio konbinazio batzuk erabiliz. Modu honetan instantzia bakoitzaren soluzio onak zein bilaketa espazioko zatietan aurki daitezkeen identifikatuko da.

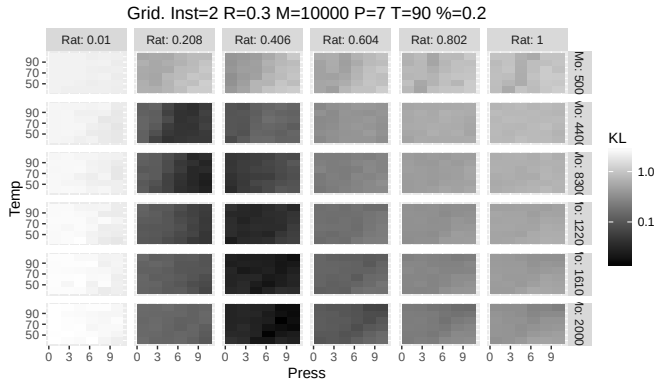
Bilaketa espazioa arakatzeko, simulazioan erabiliko diren parametro bakoitzeko 6 balio erabiliko dira. Argiago ikusteko azpian parametroek erabiliko dituzten balio guztiak ikus daitezke.

- Monomero Kopurua: $500, 4400, 8300, 12200, 16100, 20000$
- Ratioa: $0.01, 0.208, 0.406, 0.604, 0.802, 1.0$
- Presioa: $1.0, 2.8, 4.6, 6.4, 8.2, 10.0$
- Tenperatura: $40, 52, 64, 76, 88, 100$

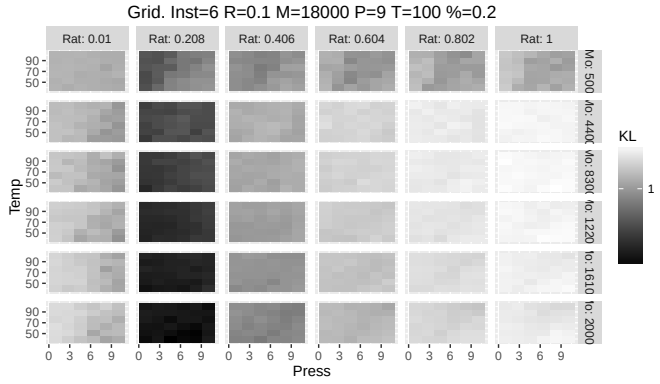
Emaitzak 4.11 grafikoan erakusten dira. Kolore argiek soluzioa txarragoa dela adierazten dute eta ilunek berriz soluzioa ona dela. Lau instantzietan ikus daiteke parametro bakoitzak duen pisua soluzio onak lortzerakoan, ratioa eta monomero kantitatea izanik pisu handiena dutenak, baina bilaketa espazioko zati handietan antzeko kolorea agertzen da, eta hori soluzio ezberdinak edukita antzeko emaitzak lortzen direnaren adierazgarri da.



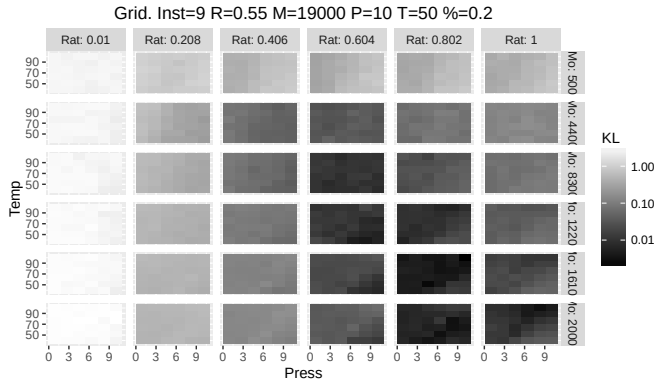
(a) 1. *Instantzia*



(b) 2. *Instantzia*



(c) 6. *Instantzia*



(d) 9. *Instantzia*

4.11. irudia: Instantzien bilaketa espazioaren arakatzea

4.3 BILAKETA ESPAZIOAREN ESPERIMENTAZIOAREN ONDORIOAK

Alde batetik, parametroen analisiari dagokionez, proben emaitzak ikusita, ratioak simulazioen emaitzetan duen pisua da deigarriena, helburu-funtzioan duen efektua handia baita. Monomero kopuruak ere badu bere garrantzia baina ratioarekin konparatuta txikia da. Presioarekin antzeko zerbait gertatzen da, baina bere efektuak denborarekiko handiagoak dira eta algoritmoen exekuzioetan ezarriko den denbora maximoa honen araberakoa izango dela erabaki da. Tenperaturari dagokionez, presioaren antzera, tenperatura optimizatzeak emaitzak hobetzeko baino, prozesua merketzeko edota azkartzeko erabiltzea hobea dela ikus daiteke.

Bestetik, lorturiko grafikoetan desbideraketa tarteetan erreparatuz ikus daiteke bi soluzio berdin simulatuta lortzen diren emaitzak oso aldakorrak izan daitezkeela. Horrek optimizazioan arazoak ekar ditzake, bi soluzioen konparaketa alda baitaiteke iterazio batetik bestera.

Hori argiago ikusteko, txatalen grafikoa egin da eta bertan bilaketa espazioko inguruko soluzio ezberdinek oso antzeko emaitzak ematen dituztela ikus daiteke. Tarte ilunek bilaketa espazioaren zati handi bat hartzen dute, eta hori, parametroek helburu-funtzioan duten pisuaren ondorioa izan daiteke, ratioak pisu handia duenez eta besteek txikiagoa, KL dibergentzian efektu txikiagoa egiten dute eta emaitzen zaraten ondorioz txatalen grafikoan espazio guztia iluna geratzen da, soluzio guztiak oso antzekoak baitira.

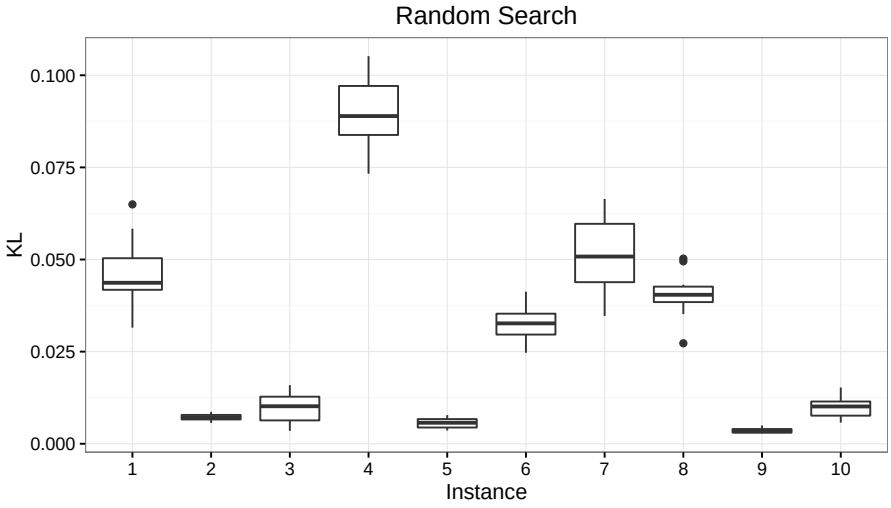
4.4 ALGORITMOEN ESPERIMENTAZIOA

Bilaketa espazioaren analisia egin ondoren, inplementatu diren hiru algoritmoekin (Ausazko bilaketa, Line Search eta Algoritmo Genetikoa) probak egin dira atal honetan. Proben bitartez, optimizazio problema mota hauetarako algoritmoek dituzten alde onak eta txarrak identifikatu eta lortutako ondorioak argitara emango dira.

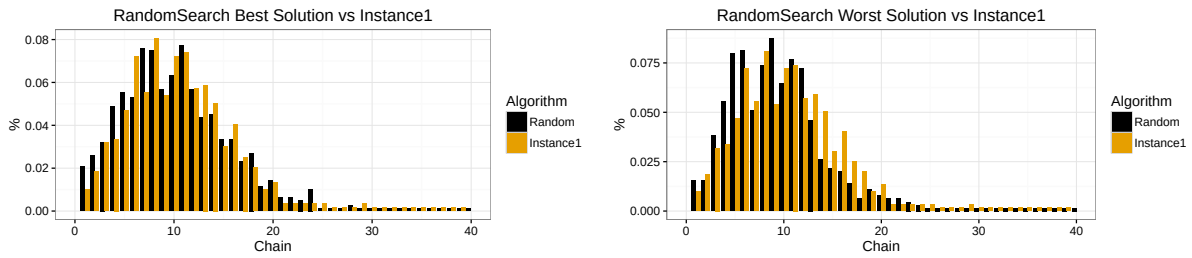
Egin diren probetan, 10 instantziak erabili dira, 0.2ko portzentaia minimoak ezarri dira eta denbora maximoa 20 segundokoa izango da.

4.4.1 Ausazko bilaketa Analisia

Ausazko bilaketa, aurretik esan bezala, oinarri moduan erabiliko den algoritmoa izango da eta beste bi optimizazio algoritmoen alde onak eta txarrak identifikatzeko balioko du. Ausazko bilaketa probatzeko, algoritmoaren 10 exekuzio egin dira instantzia bakoitzarentzat. Exekuzio bakoitzean algoritmoak 500 ebaluazio egin ditu, hau da, ausazko 500 soluzio posible aztertu ditu. Hori egin ondoren, exekuzio bakoitzeko emaitza onenak gorde dira eta 4.12 kutxa-diagraman jarri dira emaitzak. Kutxen-



4.12. irudia: Ausazko bilaketa 500 ebaluaziorekin



(a) *Ausazko Bilaketa Soluzio Onena* **(b)** *Ausazko Bilaketa Soluzio Txarrena*

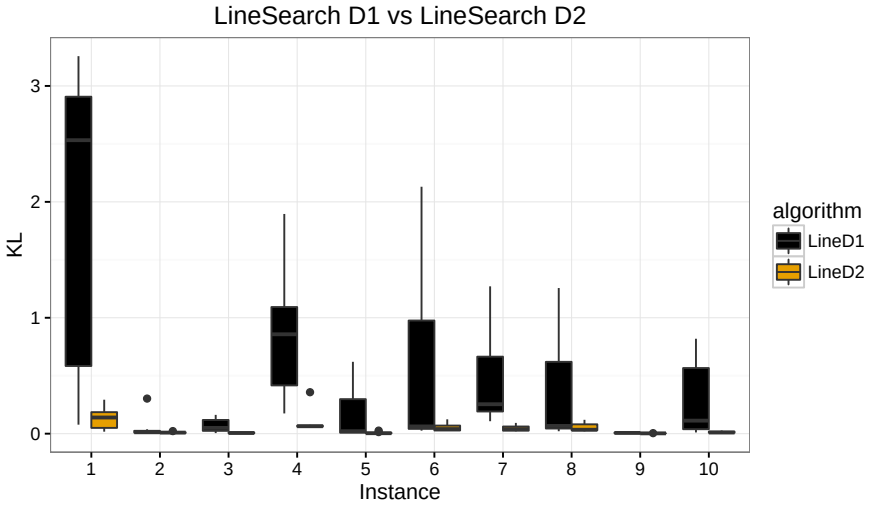
4.13. irudia: Ausazko bilaketa distribuzio onena eta txarrena vs 1. instantzia

diagrametan, instantzia bakoitzarekin egin diren 10 exekuzioetako emaitzak azaltzen dira.

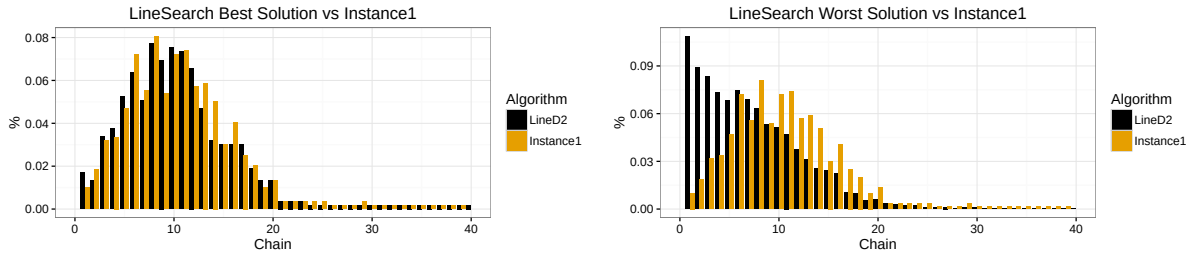
4.13 irudiko kutxa-diagraman, ausazko bilaketarekin 10 exekuzioetan lortutako distribuzio onenaren eta 1 instantziaren arteko konparaketa ikus daiteke. Bigarren kutxa-diagraman berriz, lortutako soluzio txarrenaren distribuzioa.

4.4.2 Line Search Analisia

Line Search algoritmoaren 10 exekuzio egin dira, bi distantzia erabiliz, $d_1 = (200, 0.01, 0.1, 1)$ eta $d_2 = (2000, 0.1, 1, 10)$, inguruko soluzioak aukeratzean distantzia egokia zein izan daitekeen jakiteko. Ebaluazio kopurua 500 izango da hemen ere, hiru algoritmoak baldintza berberak eduki behar baitituzte konparaketak bidezkoa izan daitezen. Hau egin ondoren, exekuzio bakoitzeko emaitza onenak gorde eta 4.14 kutxa-diagraman erakusten dira emaitzak.



4.14. irudia: Line Search Onena eta Txarrena vs 1. instantzia



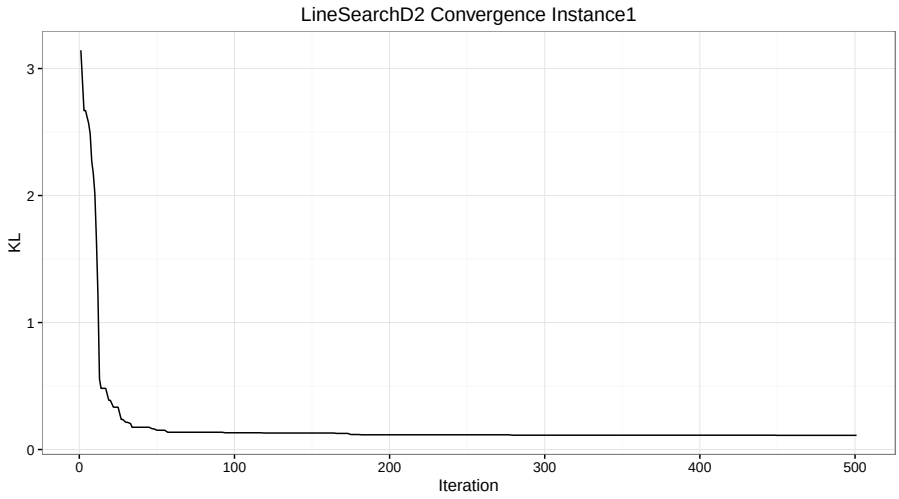
(a) *Line Search d_2 Soluzio Onena* **(b)** *Line Search d_2 Soluzio Txarrena*

4.15. irudia: LineSearch d_2 distribuzio onena eta txarrena vs 1. instantzia

4.15 irudiko lehenengo grafikoan, d_2 distantziekin lortutako algoritmoaren distribuzio onenaren eta 1 instantziaren arteko konparaketa ikus daiteke. Bigarren grafikoan berriz, d_2 distantziekin lortutako emaitza txarrenaren eta 1 instantziaren arteko distribuzioen konparaketa agertzen dira.

LineSearch Algoritmoaren Konbergentzia

Line Search algoritmok modu eraginkorrean funtzionatzen duela jakiteko, konbergentziaren analisia egin beharra dago. Analisi horrekin, algoritmoak exekuzio batean emaitzak zenbat hobe ditzakeen ikusiko da 1 instantzian. Proba hau egiteko, exekuzio batean egon diren iterazio guztietan emaitza onenek eduki duten eboluzioa zein izan den irudikatu da. 4.16 grafikoan daude iterazio bakoitzeko soluzio onenak irudikatuta.



4.16. irudia: LineSearchD1 konbergentzia 1. Instantzian

4.4.3 Algoritmo Genetikoaren Analisia

Algoritmo genetikoarekin bi proba egin dira. Lehenengoan, populazioaren tamaina egokiena zein den zehaztu da, bigarrenean berriz, populazio tamaina egokiena duen algoritmoarekin mutazio probabilitate ezberdinen analisia egin da.

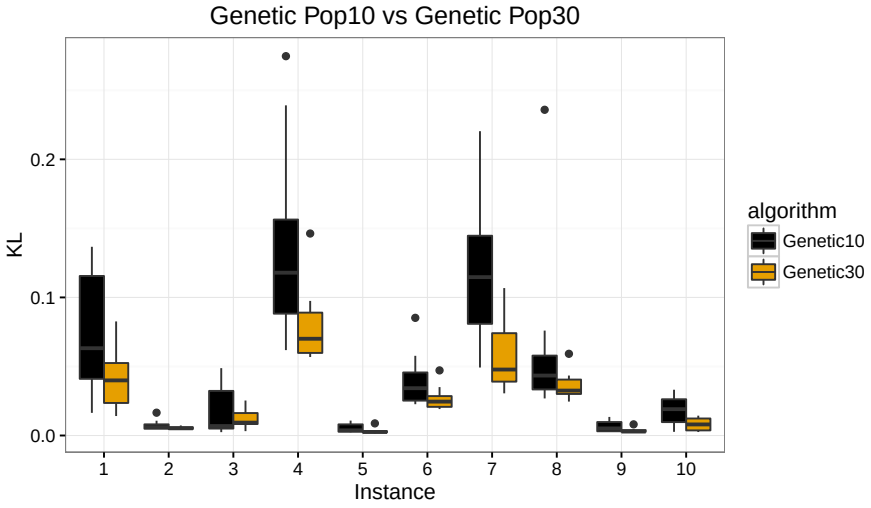
Populazio Tamaina

Beste bi algoritmoen kasuan bezala, algoritmoaren 10 exekuzio egin dira, 500 ebaluazio eginez exekuzio bakoitzean. Algoritmo genetikoa 10 eta 30eko populazioekin probatu da, dibertsitate ezberdinekin algoritmoek soluzio onak lortzeko duten kapazitatea neurtzeko. Mutazioa %0.05eko probabilitatearekin ezarri da eta belaunaldi bakoitzean sortuko diren soluzio berriak populazioaren %50 dira. Exekuzio bakoitzeko emaitza onenak gorde dira eta 4.17 kutxa-diagraman ikusi daitezke emaitzak.

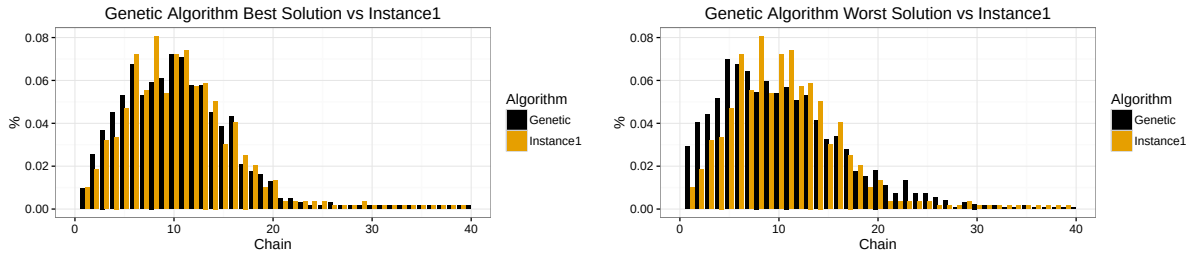
4.18 irudiko lehenengo grafikoan, 30 indibiduorekin lortutako distribuzio onenaren eta 1 instantziaren arteko konparaketa ikus daiteke. Bigarren grafikoan berriz, 30eko populazioarekin lorturiko distribuzioa txarrenaren eta 1 instantziaren artekoa.

Mutazio probabilitatea

Proba hau, mutazio balio ezberdinek algoritmo genetikoa zenbat hobe dezaketen jakiteko egin da. Mutazioaren helburua, soluzioen dibertsitatea hobetzea da helburua, belaunaldiak aurrera doazen bitartean soluzio guztiak berdinak izan ez daitezen. Probak instantzia bakoitzeko algoritmoaren 3 errepikapen eginda eta $0.05, 0.1, 0.2, 0.35, 0.5$ mutazio probabilitateekin egin dira.



4.17. irudia: Algoritmo Genetikoa Pop=10 vs Pop=30



(a) *Genetikoa Pop* = 30 *Sol. Onena*

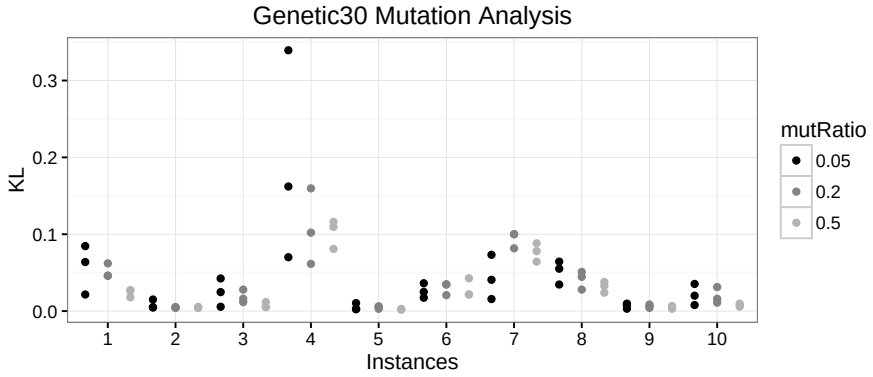
(b) *Genetikoa Pop* = 30 *Sol. Txarrena*

4.18. irudia: Algoritmo Genetikoaren distribuzio onena eta txarrena vs 1. instantzia

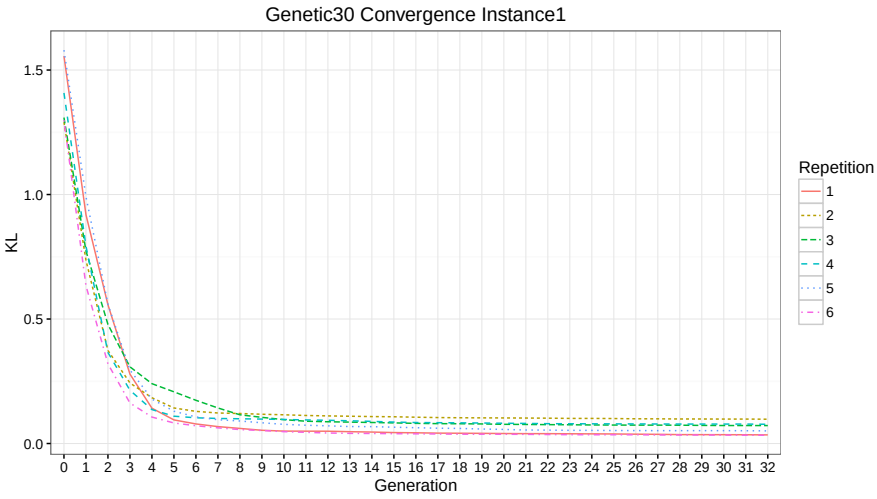
Lortutako emaitzak, 4.19 irudian ikus daitezke. Aldaketak txikiak direla ikusten da, baina mutazio probabilitateak handiak direnean soluzioa zertxobait hobetzen dela eta emaitzak ez direla hain aldakorrak ikusten da. Mutazioek parametroetan %10eko aldaketak egin ditzakete gehienez, eta mutazio altu bat jarriko bagenu hausazko bilaketa baten antzekotasuna hartuko luke. Horregatik, 0.05eko probabilitatea ezartzea erabaki da algoritmoan.

Algoritmoaren Konbergentzia

Algoritmo genetikoak modu eraginkorrean funtzionatzen duela jakiteko, konbergentziaren analisia egin beharra dago. Analisi horrekin, algoritmoak emaitzak zenbat hobe ditzakeen eta zein abiaduratan hobetzen dituen ikusiko da. Proba hau egiteko, mutazioen probabilitatea kalkulatzerakoan algoritmoaren errepikapen bakoitzean ebaluatu diren soluzioen emaitzak gorde dira eta ondoren, belaunaldiz belaunaldi eduki duten eboluzioa zein izan den irudikatu da. 4.20 grafikoan daude belaunaldi bakoitzeko po-



4.19. irudia: Algoritmo genetikoak mutazio ezberdinekin



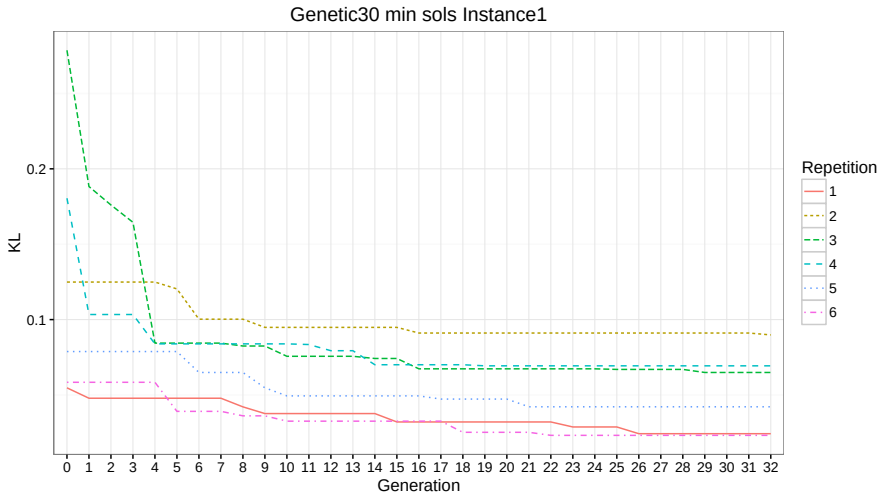
4.20. irudia: Genetikoaren konbergentzia 1. Instantzian

pulazioaren batezbestekoen konbergentzia. 4.21 irudian berriz, belaunaldi bakoitzean lortutako emaitza onenak agertuko dira, emaitza onena zenbat hobetzen den ikusteko.

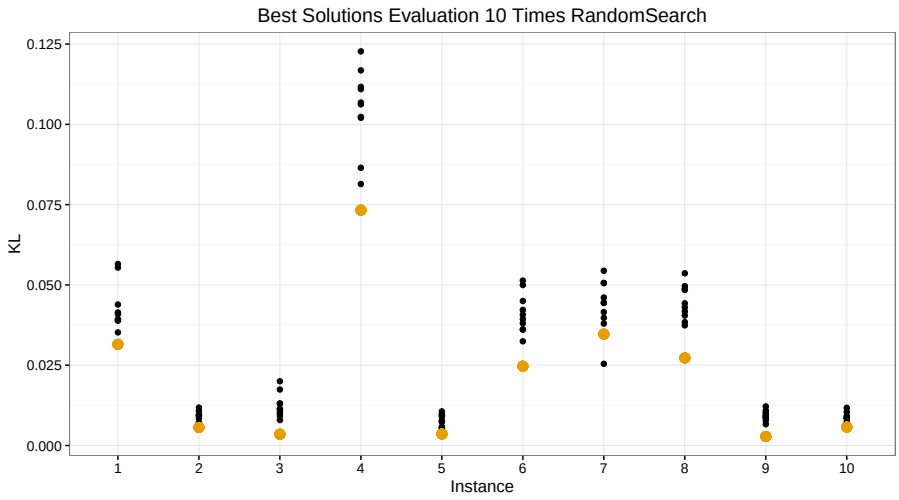
Emaitzen Aldakortasuna

Soluzio bera behin baino gehiagotan exekutatuta emaitzak zenbat aldatzen diren ikusteko, Ausazko bilaketa, Line Search eta Algoritmo genetikoa algoritmoek instantzia bakoitzean lortu dituzten soluzio onenen 10 exekuzio egin dira.

4.22 irudian ausazko bilaketa algoritmoaren emaitzen aldakortasunaren analisia agertzen zaigu, 4.23 grafikoan Line Search algoritmoarena eta 4.24 irudian 30eko populazio duen algoritmo genetikoarena.



4.21. irudia: Genetikoaren soluzio onenak 1. Instantzian



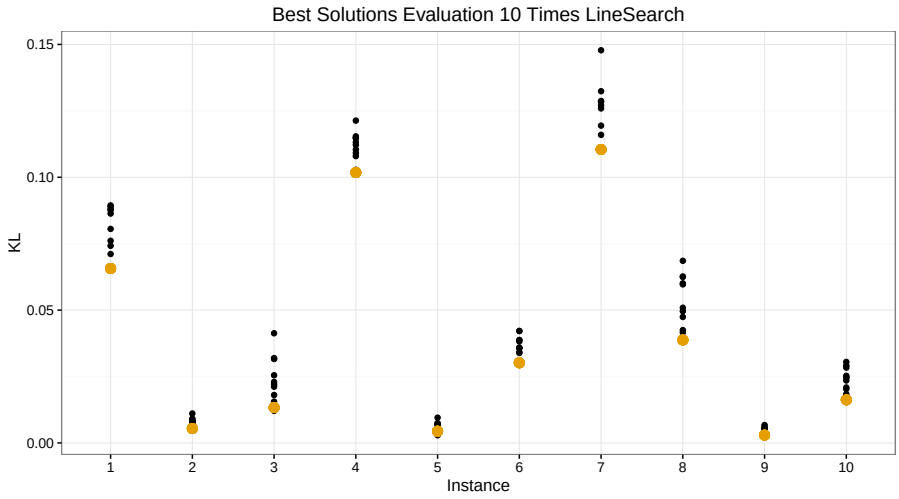
4.22. irudia: Ausazko Bilaketa Aldakortasun analisia

Algoritmoen Konparaketa

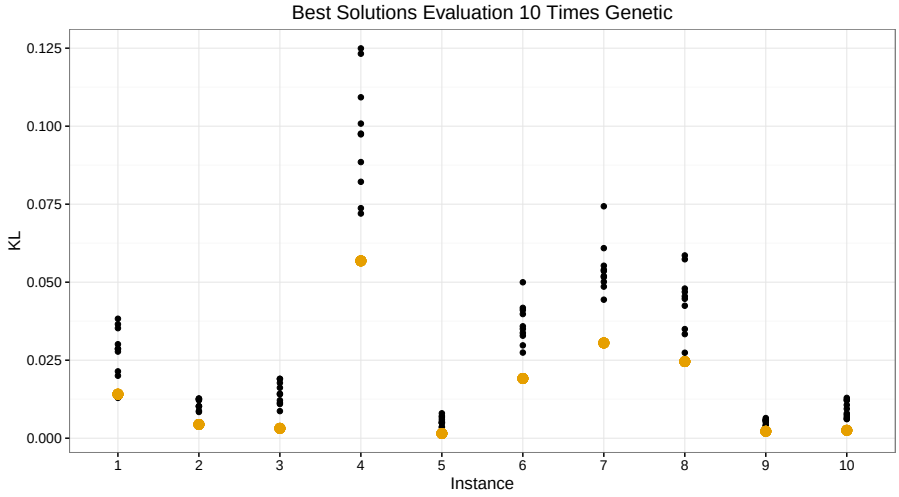
Azkenik, hiru algoritmoek lortutako soluzioen artean konparaketak egiteko, 4.25 grafikoa sortu da.

4.5 ALGORITMOEN ESPERIMENTAZIOAREN ONDORIOAK

Proben emaitzak ikusi ondoren, inplementatutako algoritmo metaheuristikoek problema mota hauetan dituzten alde txarrak eta onak azalduko dira.



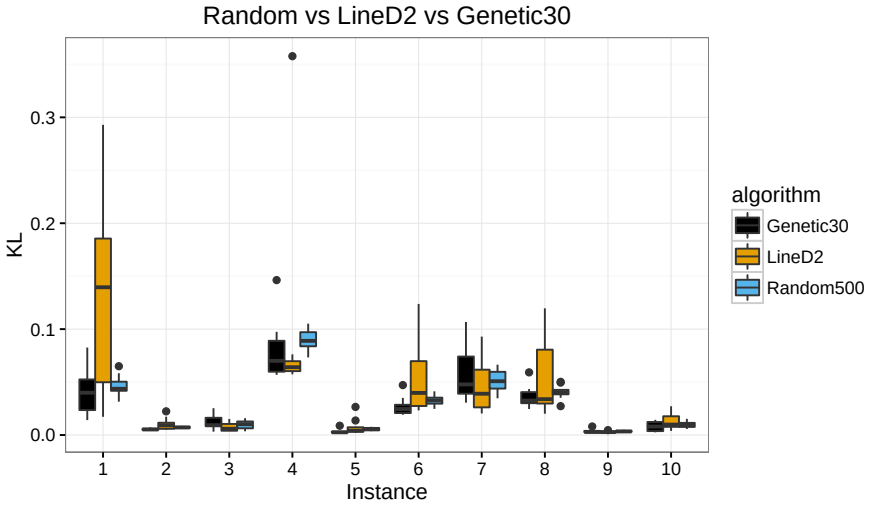
4.23. **irudia:** Line Search Aldakortasun analisia



4.24. **irudia:** Algoritmo Genetikoa Aldakortasun analisia

Line Search algoritmoari dagokionez, minimo lokalak bilatzeko gaitasun handia duela ikusi da, oso iterazio gutxitan emaitza asko hobetzen baitu, baina, lortzen duen emaitza ez da izaten beti ona, hasieran aukeratutako puntuaren arabera soluzio onak edo txarrak lortuko baititu. Horren ondorioz, hiru algoritmoetatik emaitza aldakorrenak lortzen dituena da (ikusi 4.25 irudia). Aldagaiak banan-bana hartuta helburu-funtzioa konbexua dela ikus zitekeen arren aurreko atalean eta Line Search algoritmoek problema konbexuetan onak direla ikusita algoritmoaren errendimendu txarra arrazoi hauengatik izan daitekeela ondorioztatu da:

- Parametro bakarra optimizatzen duenez iterazio bakoitzean, besteak ez ditu kontuan hartzen eta ondorioz, soluzioak ez dira nahi bezala hobetzen.



4.25. irudia: Algoritmo guztien konparaketa

- Finkatu diren d distantziak egokiak izan gabe jarraitzen dute eta ondorioz inguruko soluzioetan ez da hobekuntzarik lortzen, egiten diren pausoak motzegiak edo luzeegiak direlako.
- Simulatzailearen aldakortasunaren ondorioz zarata sortzen da inguruko soluzioetan eta nahiz eta soluzio hobea egon, trabaturik gera daiteke.

Algoritmo genetikoaren kasuan berriz, hasieran soluzio bat baino gehiago dituenez, soluzio ezberdinak lortzen dira. Populazioan dibertsitatea beharrezkoa denez, 30 indibiduoko populazioa duen algoritmoa egokiagoa izateak zentzua du, hasieran soluzio gehiago daudenez, dibertsitatea handiagoa da eta soluzio on batetik hasteko probabilitatea handitzen da.

Soluzio on batetik asten den arren, algoritmo genetikoak ez du asko hobetzen lortzen den emaitza. Dena dela, horrek zentzua du, emaitzak gutxi hobetzen diren arren askoz ere emaitza hobeak lortzea zaila baita erabiltzen den simulatzailearen ondorioz. Simulatzailearen emaitzak oso aldakorrak dira eta ezin izan da ikusi soluzio hobeak lortzeko gaitasuna edukiko duen ala ez. Halere, algoritmoaren konbergentzia eta soluzio onenen hobekuntza tarte txikiak ikusita, ez du ematen algoritmo egokiena izan daitekeenik soluzio onenak bilatzeko.

Azkenik, emaitzen aldakortasunaren analisi bat egin da eta bertan argi ikus daiteke algoritmoek bilatzen dituzten soluzioak normalean lortu ahal diren emaitza onenak direla eta ondorioz, soluzio bera exekutatuko bagenu soluzio okerragoak lortuko liratekeela.

5 ONDORIOAK

Proiektuan polimerizazio-prozesuetako erreakzio-baldintzen optimizazio problemen analisi bat egin da eta algoritmo metaheuristikoek izan ditzaketen alde onak eta txarrak aztertu dira. Problema mota horiek nola analizatu eta ebatzi daitezkeen jakiteko, proiektuaren ezaugarrietara egokituko den simulatzaile bat inplementatu, bere bilaketa espazioa analizatu eta bi algoritmo metaheuristikoren (Line Search eta Algoritmo Genetikoa) inplementazioa eta analisia egin dira proiektu honetan.

Inplementatu den simulatzailea, erreala ez den simulatzaile bat izan da, proiektuaren helburua problema hauek nola ebatz daitezkeen eta metaheuristikoen egokitasuna ebaluatzea izanik, problema erreal baten aurrean jartzea konplexuegia izango litzatekeelako gradu amaierako proiektu batean, denbora eta konputazio ahalmenak mugatuak baitira. Halere, simulatzailearen aukeraketa egitean zeuden aukerak urrien ondorioz, inplementatutako simulatzaileak gabeziak izan ditu, eta horietako batzuk gainera analisi hobeak egitea baldintzatu gaituzte.

Bilaketa espazioaren analisiak, problema mota hauetan analisi zuzen bat nola egin daitekeen eta bilaketa espazioa analizatzea ondorioak ateratzeko zein erabilgarria izan daitekeen ikusi da. Gainera, inplementatuko diren optimizazio-algoritmoek nola funtziona dezaketenaren ideia bat ematen du eta beraz, algoritmo egokiak inplementatzeko analisi hau egiteak duen garrantzia azpimarratu nahi da.

Optimizazio-algoritmoetan, simulatzaileen emaitzen aldakortasunak ekar ditzakeen arazoak konpontzeko oso konplexuak direla kontura gaitezke, zaila baita polimero distribuzio berdin berdina lortzea soluzio berdinetatik hasita ere. Gure simulatzailearen kasuan aldakortasuna handia izan da, baina simulatzaile errealago batean zarata txikiagoko emaitzak lortzea errazagoa izango litzateke. Horrek, denborarekin ere badu erlazioa, zarata gutxiko simulazioek denbora kostu handiak izaten baitituzte, eta honek mugak jar ditzake konputazio ahalmen aldetik.

Simulatzaile errealez hitz egiten jarraituz, bi eredu mota daude: eredu deterministak eta estokastikoak. Sarreran aipatu den moduan, gaur egun erabilienak estokastikoak dira eta eginiko proiektua eredu horietara zuzendu da. Estokastikoek emaitzak aldakorrak izaten dituzte eta, horrek, inguruko soluzioek emaitzak hobetzen dituzten ala ez ikustea konplikatzen du, puntu batetik aurrera soluzio hobeak lortzea ezinezkoa eginez.

Proiektuan ikusitako gabeziak konpontzeko, etorkizunean egin daitekeen lan ugari dago. Lehenengoetako bat, simulatzaile estokastiko erreal bat lortzea izango litzateke, proiektuaren diru eta konputazio kostuak asko handitu arren problema erreal bat ebazteko guztiz beharrezkoa baita. Simulatzaile estokastikoek, konputazio ahalmena

txikiagotzearen truke, emaitzetan aldakortasuna sortzen dutenez, interesgarria izango litzateke aldakortasuna ahal den hobekien kudeatu dezaketen tekniken erabilera egitea.

Optimizazio algoritmoetan zentratuz, etorkizunean simulatzaileetan aldakortasun edota zarata honekin jokatu daiteke simulazio azkarragoak eginez eta emaitzen aldakortasun handiagoarekin jokatuz soluzio onak bilaketa espazioko zein tokitan egon daitezkeen jakiteko, 4.2.1 puntuan egin den antzeko zerbait eginez. Konputazio kostu txikiagoko bilaketa globalak egin ondoren, soluzio onen inguruetan zarata gutxiago duten simulazioekin optimizazio-algoritmo lokalen erabilera egitea ideia ona izan liteke eta soluzio on bat lortzera eraman dezake. Beste optimizazio-algoritmo batzuen erabilera egitea ere interesgarria izan liteke, naturan oinarritutako Particle Swarm Inteligence (PSO) [Kennedy, 2010], motako metaheuristikoena erabilera egitea adibidez.

Proiektu honekin beraz, etorkizunean simulatzaile erreal bat eta metaheuristikoen erabilera egingo duen edozein ikerketarentzat oinarri bat ezarri nahi izan da. Proiektua, antzeko problema baten aurrean izanda sor daitezkeen arazoak modu errazagoan identifikatzeko eta metaheuristiko egokiak bilatzeko lagungarria izan daitekeela uste dugu. Halere, azpimarratu behar da proiektuan zoritxarrez ezin izan dela metaheuristiko egoki bat bilatu eta simulatzaile errealen erabilera egin ez denez beti mugatuko du proiektu konplexu batean egon daitezkeen arazoak identifikatzerakoan.

BIBLIOGRAFIA

[Blum and Roli, 2003] Blum, C. and Roli, A. (2003). Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Comput. Surv.*, 35(3):268–308.

[Fernandes et al., 2004] Fernandes, F. A., Lona, L. M., and Penlidis, A. (2004). Inverse modeling applications in emulsion polymerization of vinyl acetate. *Chemical Engineering Science*, 59(15):3159 – 3167.

[Gaillard et al., 2009] Gaillard, F., Kubera, Y., Mathieu, P., and Picault, S. (2009). *Engineering Societies in the Agents World IX: 9th International Workshop, ESAW 2008, Saint-Etienne, France, September 24-26, 2008, Revised Selected Papers*, chapter A Reverse Engineering Form for Multi Agent Systems, pages 137–153. Springer Berlin Heidelberg, Berlin, Heidelberg.

[García et al., 2008] García, M. J., Boulanger, P., and Henao, M. (2008). Structural optimization of as-built parts using reverse engineering and evolution strategies. *Structural and Multidisciplinary Optimization*, 35(6):541–550.

[Gooch, 2011] Gooch, J. W. (2011). *Encyclopedic Dictionary of Polymers*, chapter Free-Radical Polymerization, pages 326–326. Springer New York, New York, NY.

[Hamzehlou et al., 2014] Hamzehlou, S., Ballard, N., Carretero, P., Paulis, M., Asua, J. M., Reyes, Y., and Leiza, J. R. (2014). Mechanistic investigation of the simultaneous addition and free-radical polymerization in batch miniemulsion droplets: Monte carlo simulation versus experimental data in polyurethane/acrylic systems. *Polymer*, 55(19):4801 – 4811.

[Hemker, 2010] Hemker, T. (2010). *Derivative Free Surrogate Optimization for Mixed-Integer Nonlinear Black Box Problems in Engineering*. TU Darmstadt, Darmstadt. Druckausgabe: Düsseldorf, VDI-Verl., 2009. ISBN 978-3-18-379710-3 (Fortschritt-Berichte VDI, R. 10, Bd. 797) [Darmstadt, TU, Diss. 2008].

[Kennedy, 2010] Kennedy, J. (2010). *Particle Swarm Optimization*, pages 760–766. Springer US, Boston, MA.

[Kullback and Leibler, 1951] Kullback, S. and Leibler, R. A. (1951). On information and sufficiency. *Ann. Math. Statist.*, 22(1):79–86.

[Mohammadi et al., 2014] Mohammadi, Y., Pakdel, A. S., Saeb, M. R., and Boodhoo, K. (2014). Monte carlo simulation of free radical polymerization of styrene in a spinning disc reactor. *Chemical Engineering Journal*, 247:231 – 240.

[Nascimento et al., 2000] Nascimento, C. A. O., Giudici, R., and Guardani, R. (2000). Neural network based approach for optimization of industrial chemical processes.

Computers & Chemical Engineering, 24(910):2303 – 2314.

[Rios and Sahinidis, 2013] Rios, L. M. and Sahinidis, N. V. (2013). Derivative-free optimization: a review of algorithms and comparison of software implementations. *Journal of Global Optimization*, 56(3):1247–1293.

[Sivanandam and Deepa, 2008] Sivanandam, S. and Deepa, S. (2008). *Introduction to Genetic Algorithms*, chapter Genetic Algorithms, pages 15–37. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Snyman, 2005] Snyman, J. A. (2005). *Line Search Descent Methods for Unconstrained Minimization*, pages 33–55. Springer US, Boston, MA.

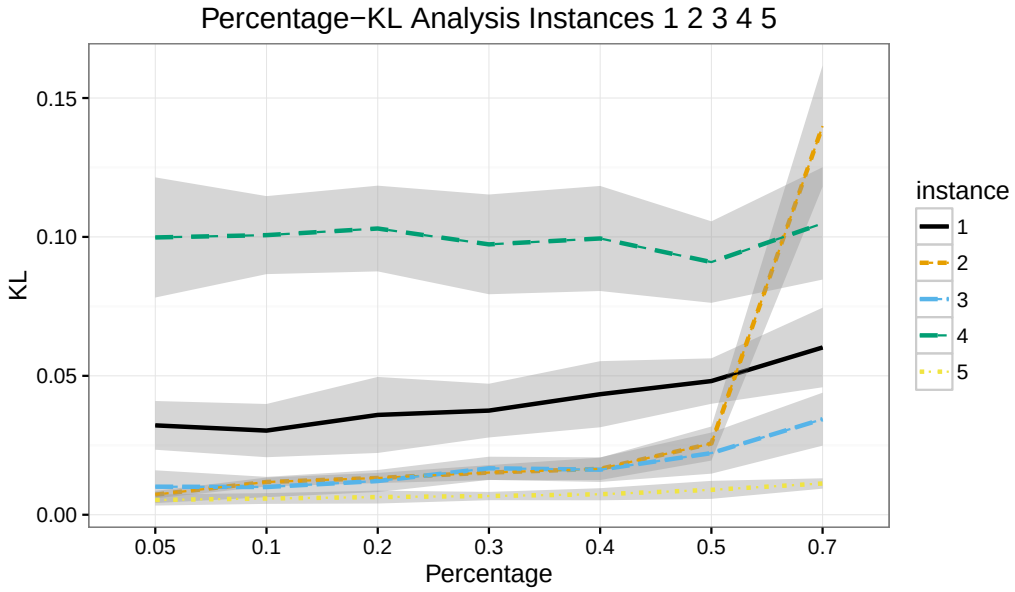
[Wilensky, 1998] Wilensky, U. (1998). Netlogo radical polymerization model. http://ccl.northwestern.edu/netlogo/models/radicalpolymerization, Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL.

[Wilensky, 1999] Wilensky, U. (1999). Netlogo. http://ccl.northwestern.edu/netlogo/, Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL.

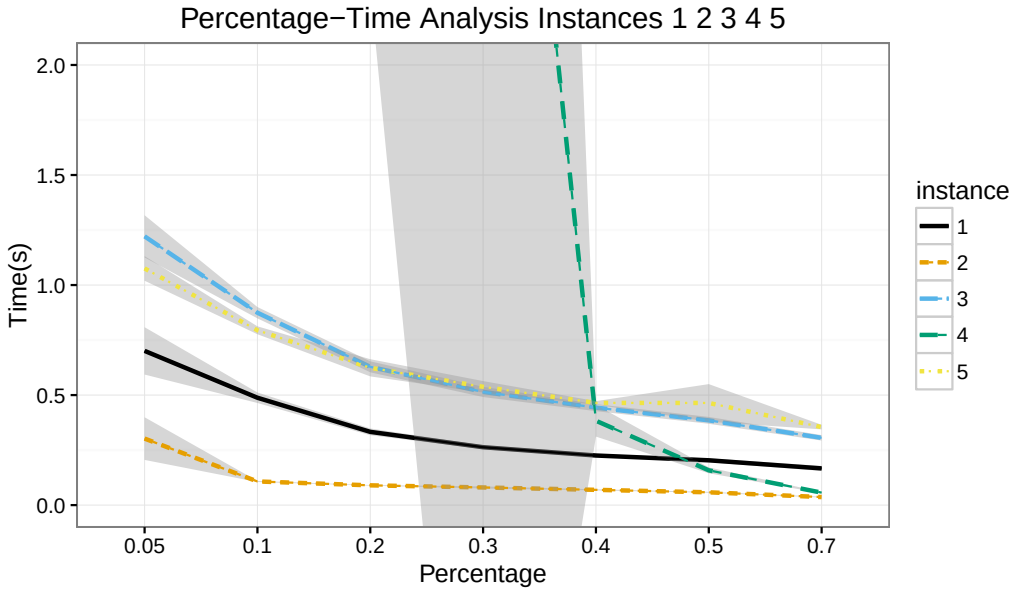
[Zabell, 1989] Zabell, S. L. (1989). The rule of succession. *Erkenntnis*, 31(2):283–321.

6 ERANSKINAK

Eranskinetan bilaketa espazioaren esperimentazioan azaldu ez diren instantzia guztien analisiak azalduko dira.

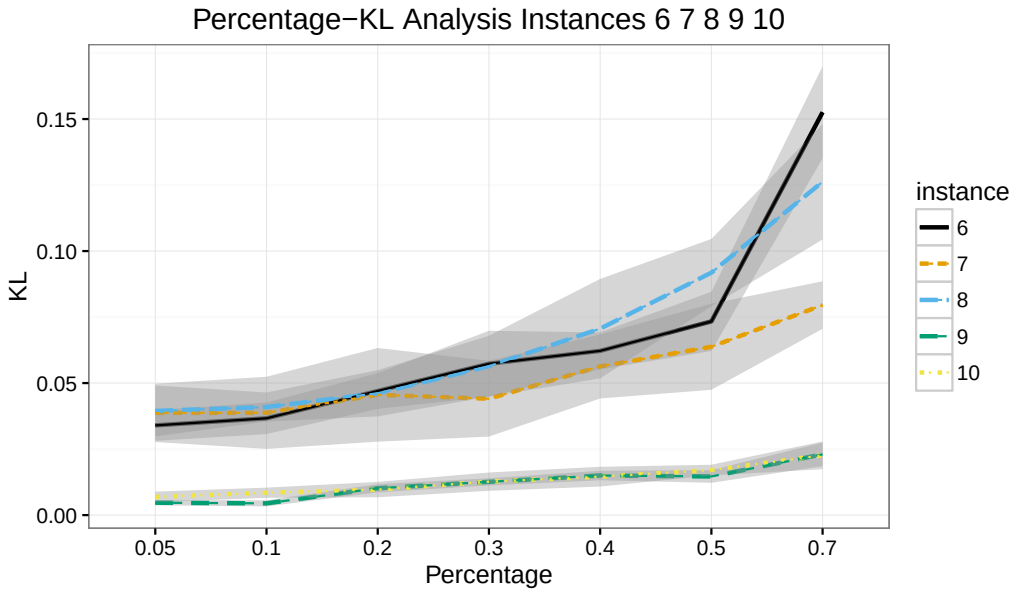


(a) *KL dibergentzia*

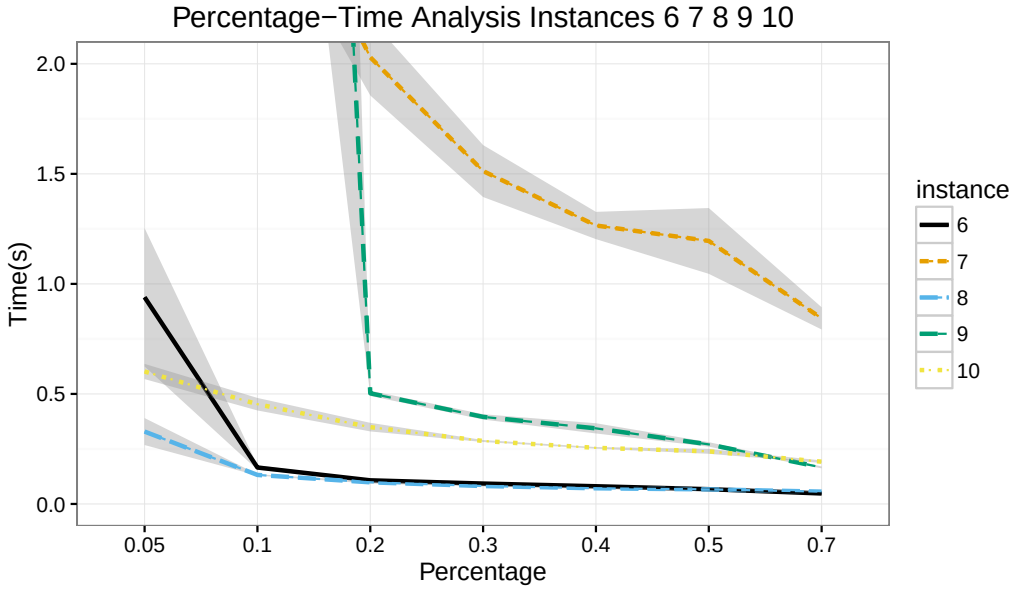


(b) *Exekuzio − Denborak*

6.26. irudia: Portzentaia Minimoen Analisia 1, 2, 3, 4 eta 5 Instantziak



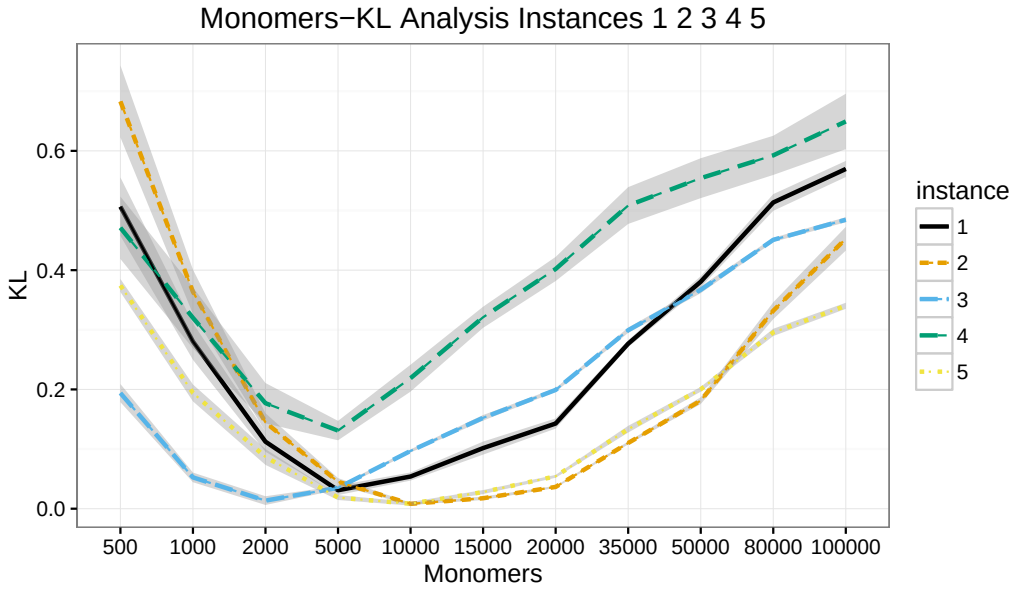
(a) *KL dibergentzia*



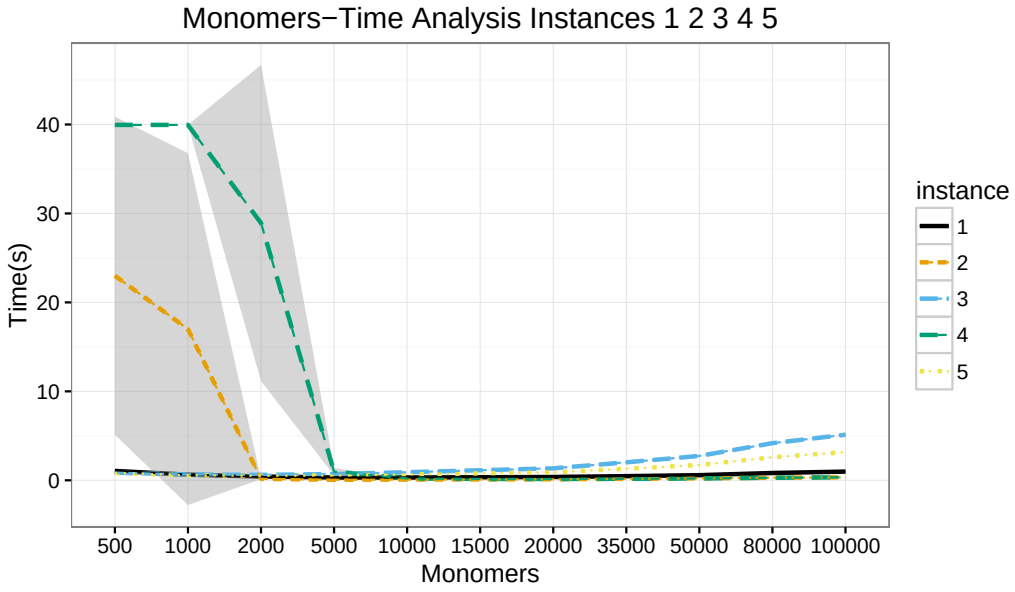
(b) *Exekuzio − Denborak*

6.27. irudia: Portzentaia Minimoen Analisia 6, 7, 8, 9 eta 10 Instantziak

6.2 MONOMEROEN ANALISIA

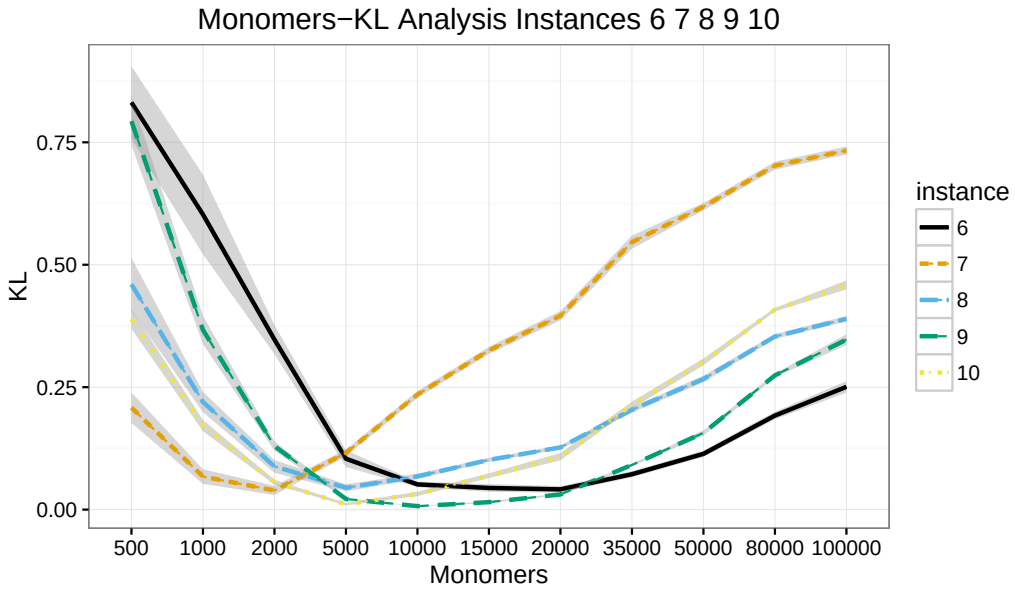


(a) *KL dibergentzia*

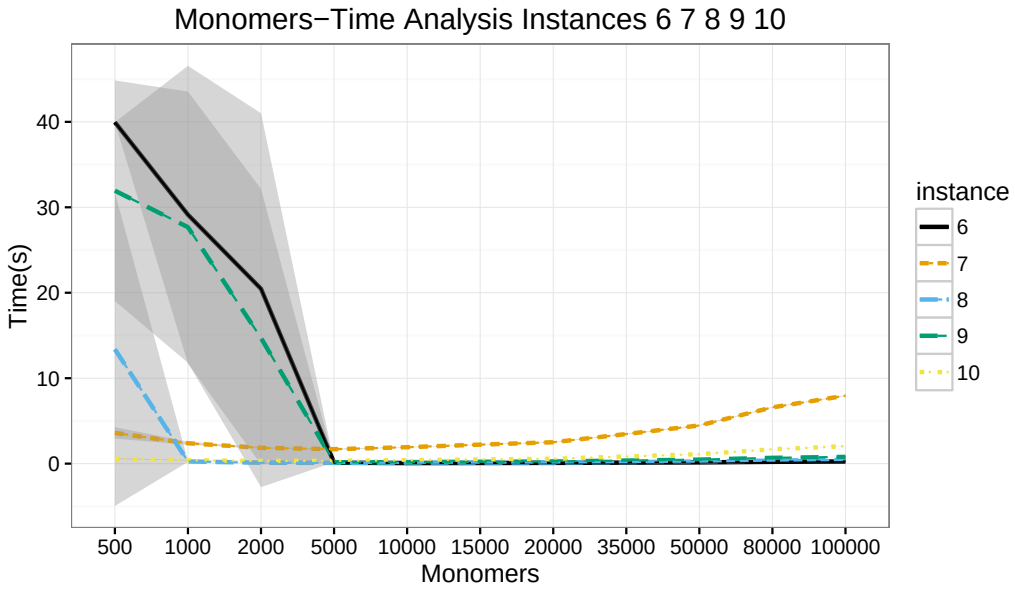


(b) *Exekuzio − Denborak*

6.28. irudia: Monomeroen Analisia 1, 2, 3, 4 eta 5 Instantziak



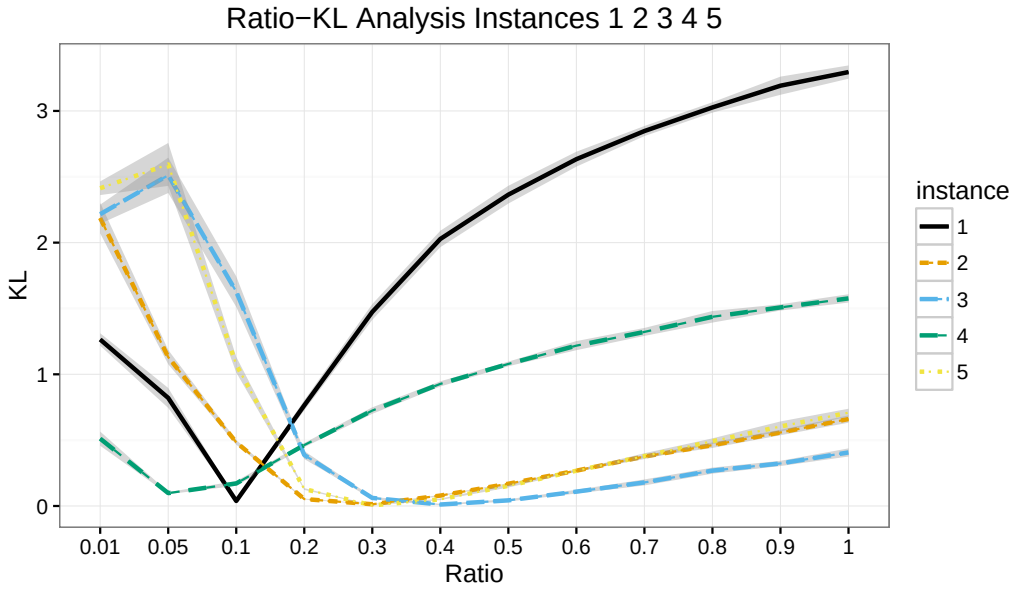
(a) *KL dibergentzia*



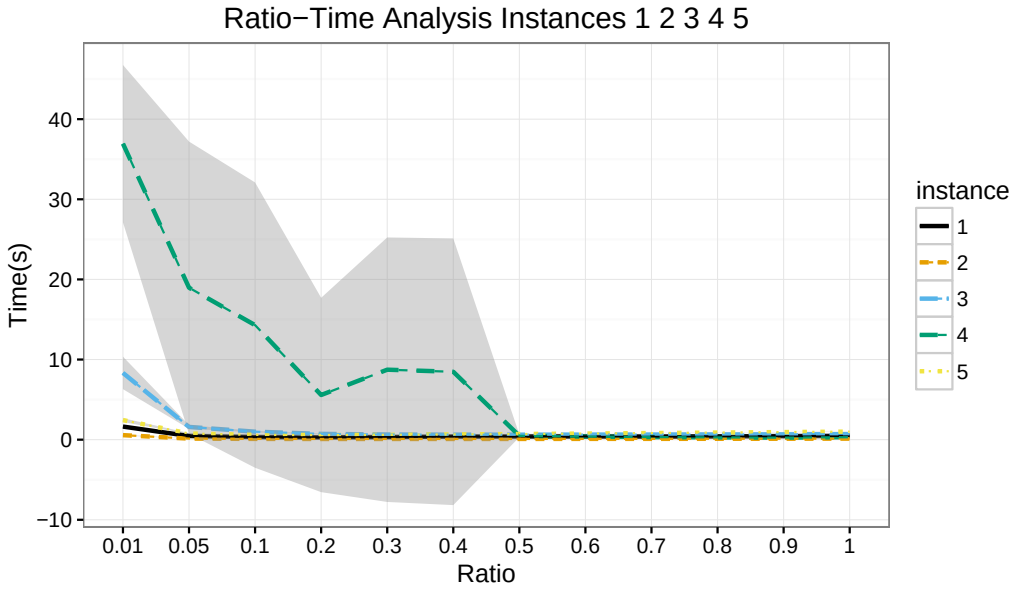
(b) *Exekuzio − Denborak*

6.29. irudia: Monomeroen Analisia 6, 7, 8, 9 eta 10 Instantziak

6.3 RATIOAREN ANALISIA

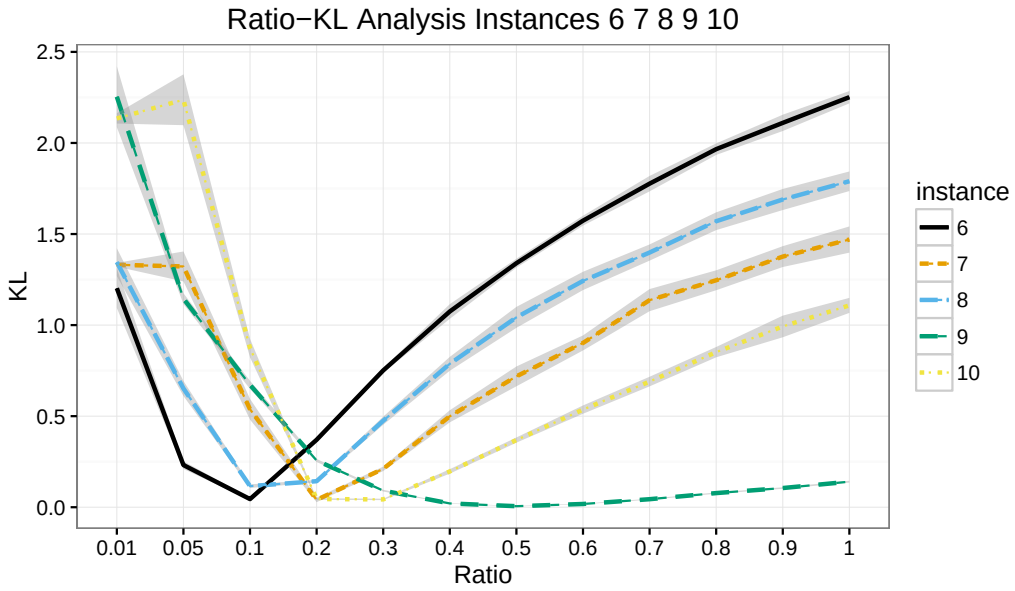


(a) *KL dibergentzia*

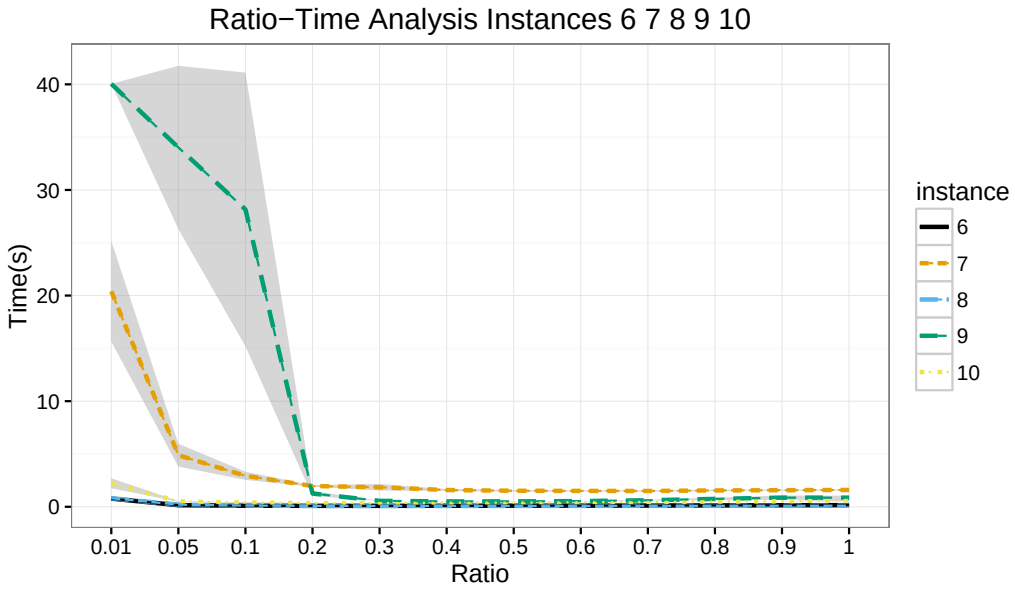


(b) *Exekuzio – Denborak*

6.30. irudia: Ratioaren Analisia 1, 2, 3, 4 eta 5 Instantziak



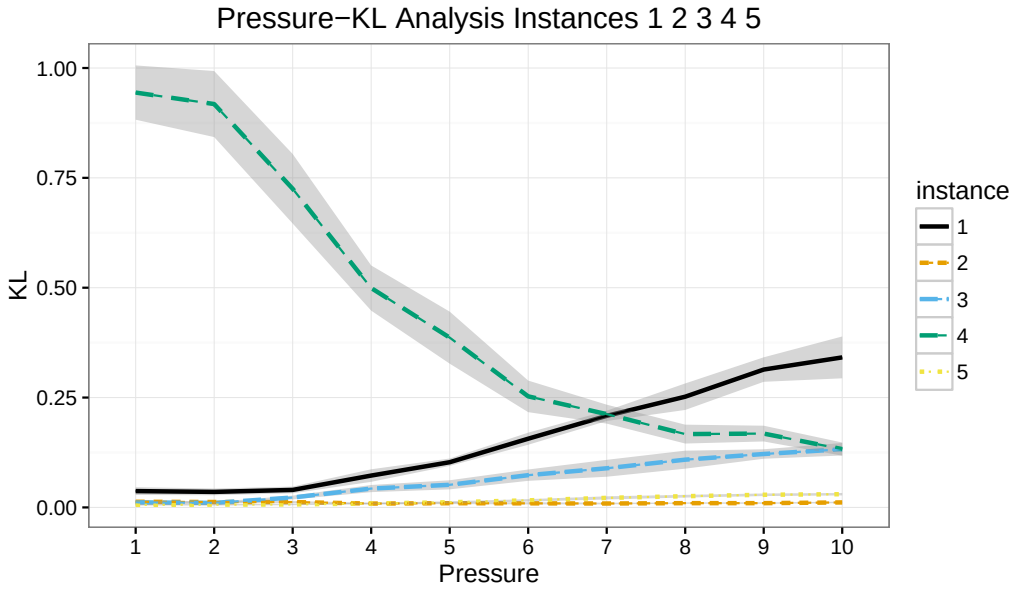
(a) *KL dibergentzia*



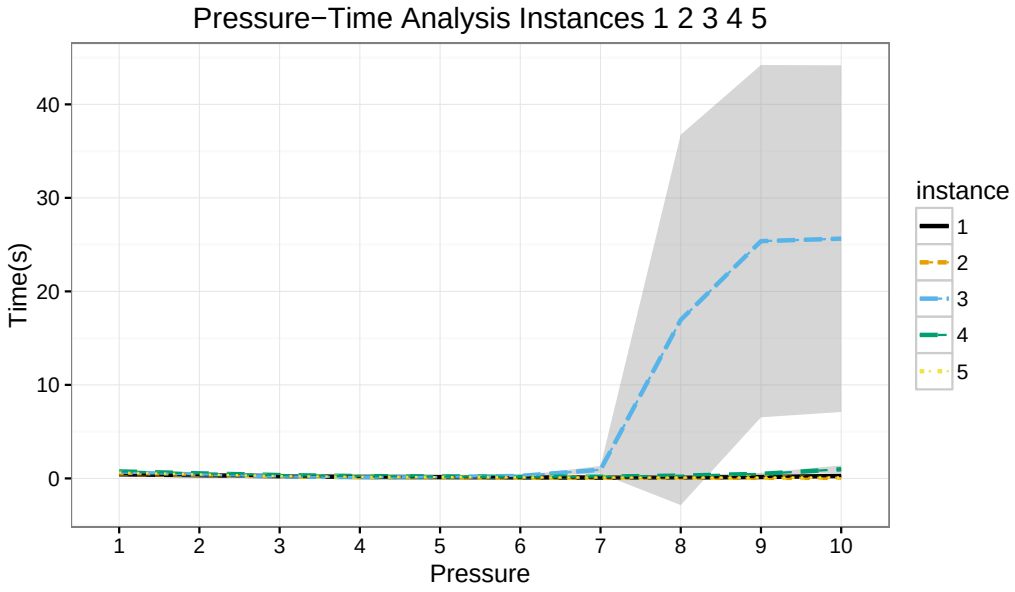
(b) *Exekuzio − Denborak*

6.31. irudia: Ratioaren Analisia 6, 7, 8, 9 eta 10 Instantziak

6.4 PRESIOAREN ANALISIA

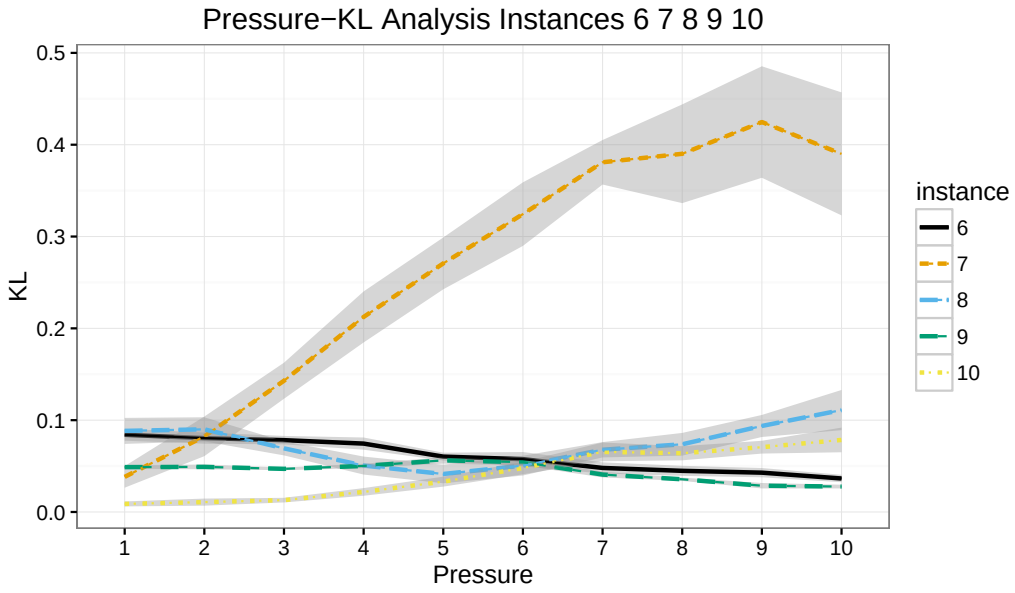


(a) *KL dibergentzia*

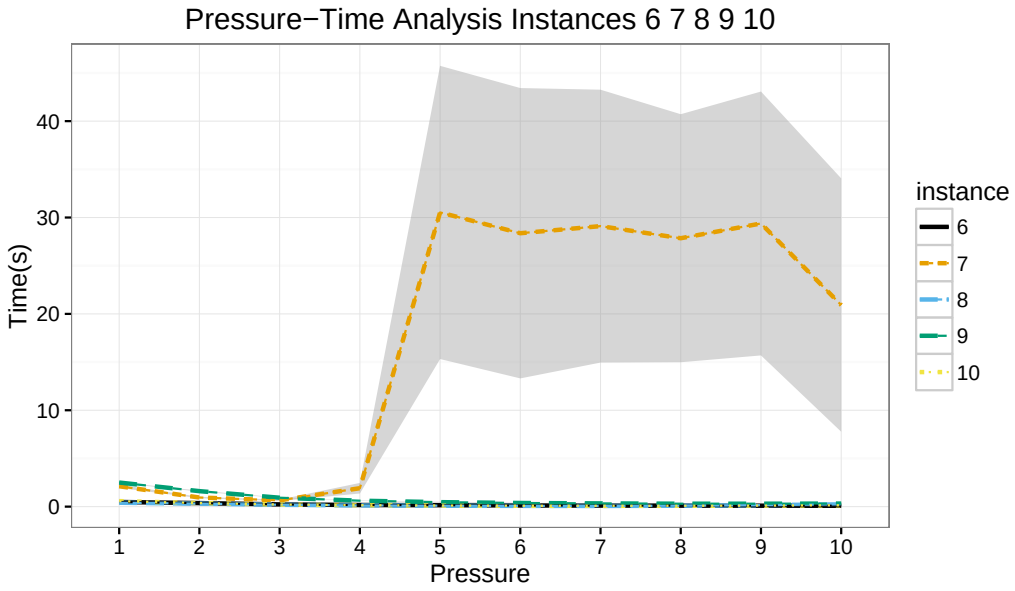


(b) *Exekuzio − Denborak*

6.32. irudia: Presioaren Analisia 1, 2, 3, 4 eta 5 Instantziak



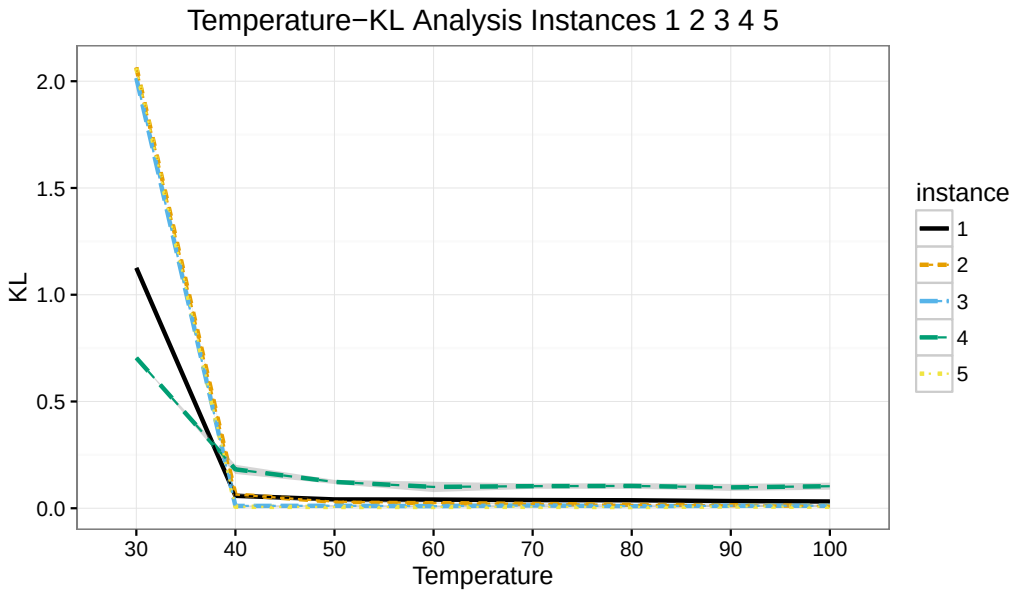
(a) *KL dibergentzia*



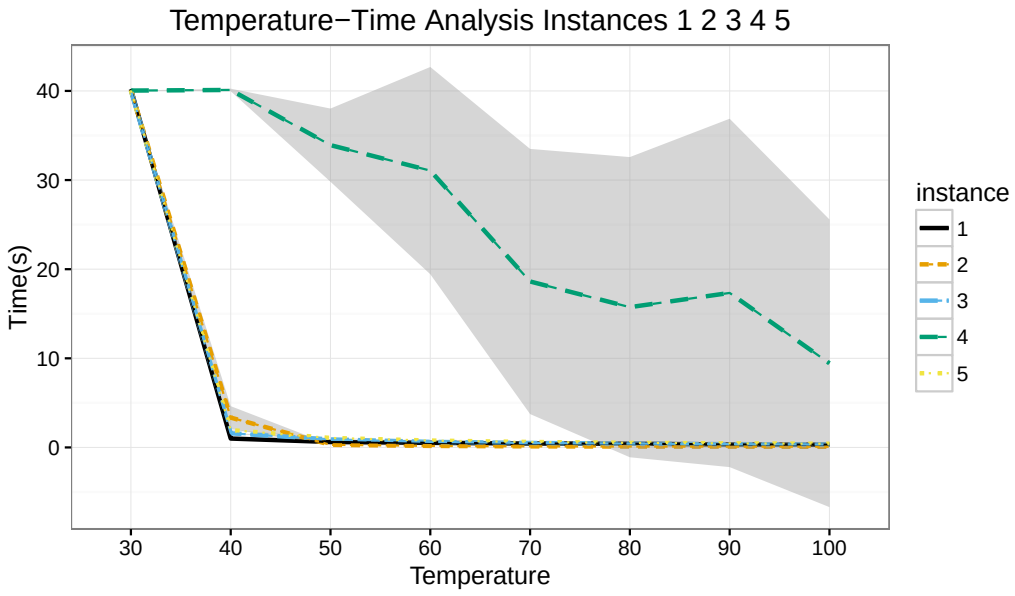
(b) *Exekuzio − Denborak*

6.33. irudia: Presioaren Analisia 6, 7, 8, 9 eta 10 Instantziak

6.5 TENPERATURAREN ANALISIA

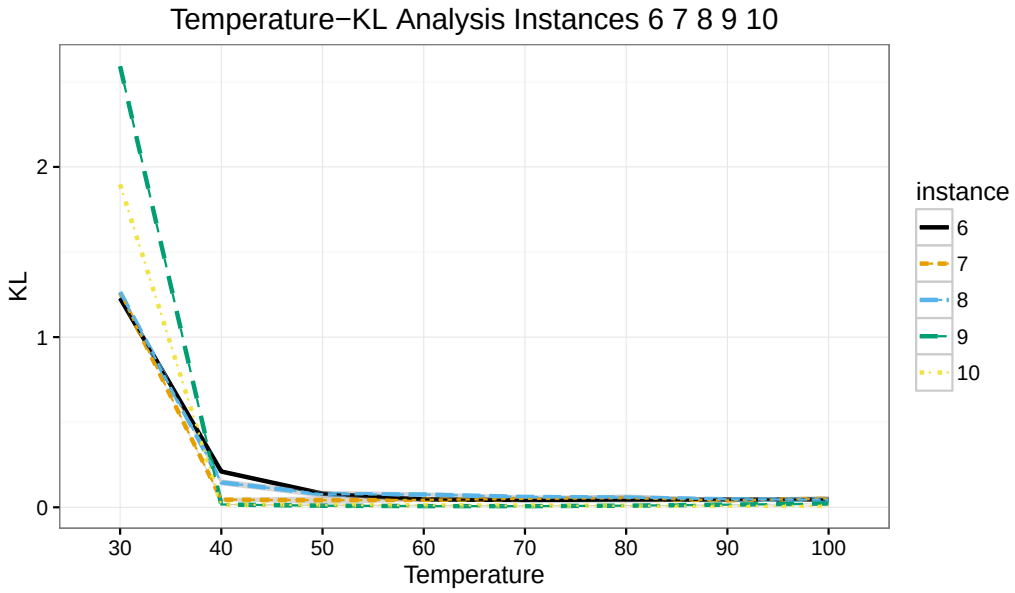


(a) *KL dibergentzia*

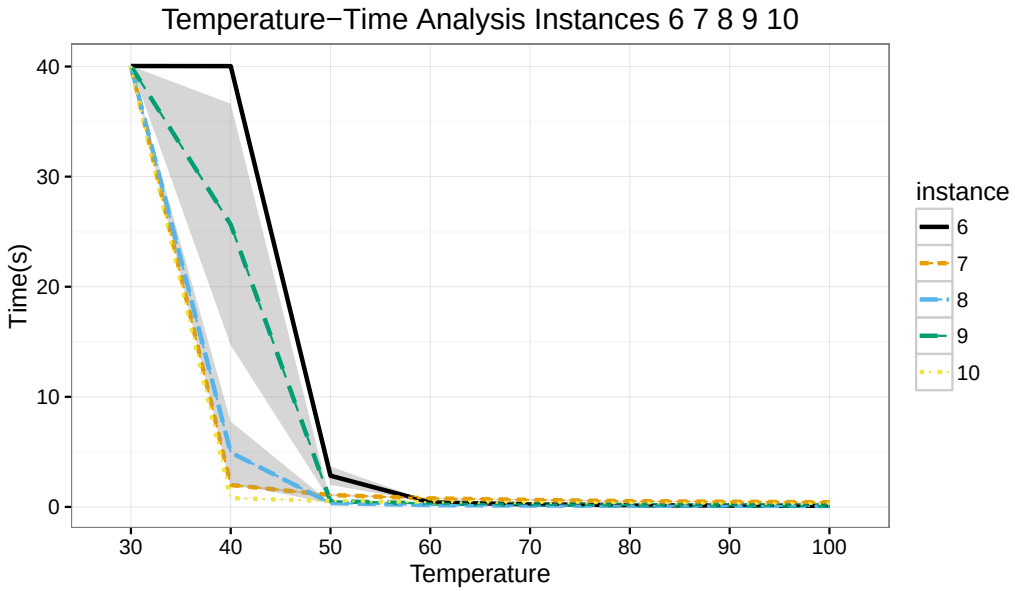


(b) *Exekuzio − Denborak*

6.34. irudia: Tenperaturaren Analisia 1, 2, 3, 4 eta 5 Instantziak

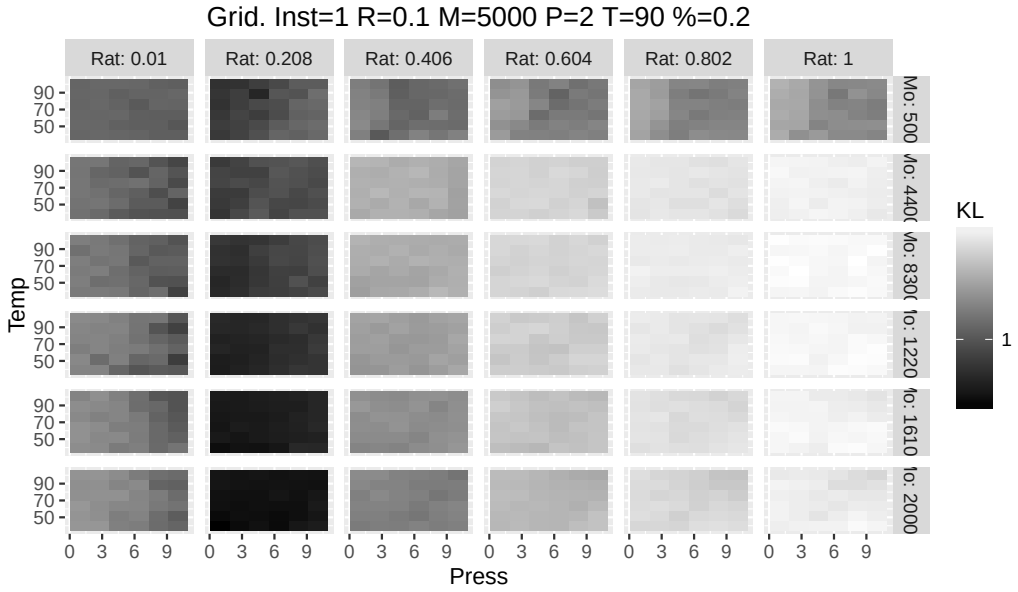


(a) *KL dibergentzia*

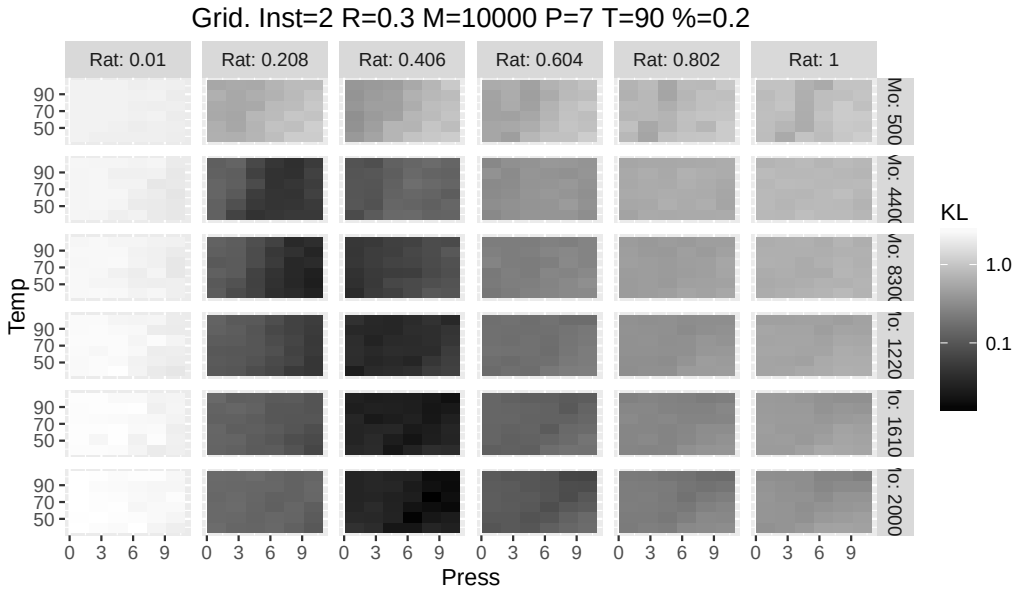


(b) *Exekuzio − Denborak*

6.35. irudia: Tenperaturaren Analisia 6, 7, 8, 9 eta 10 Instantziak

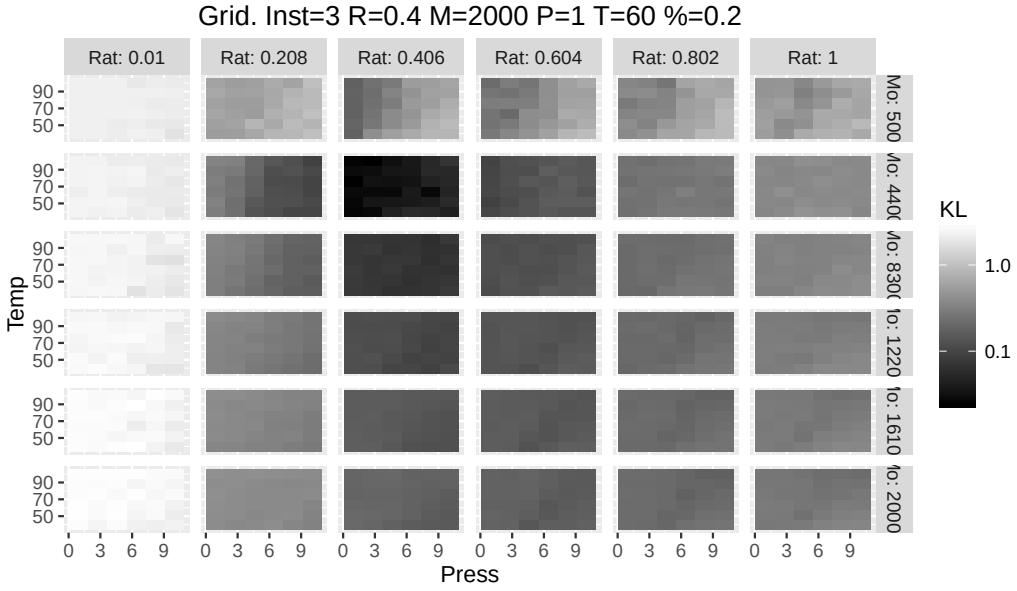


(a) 1 *Instantzia*

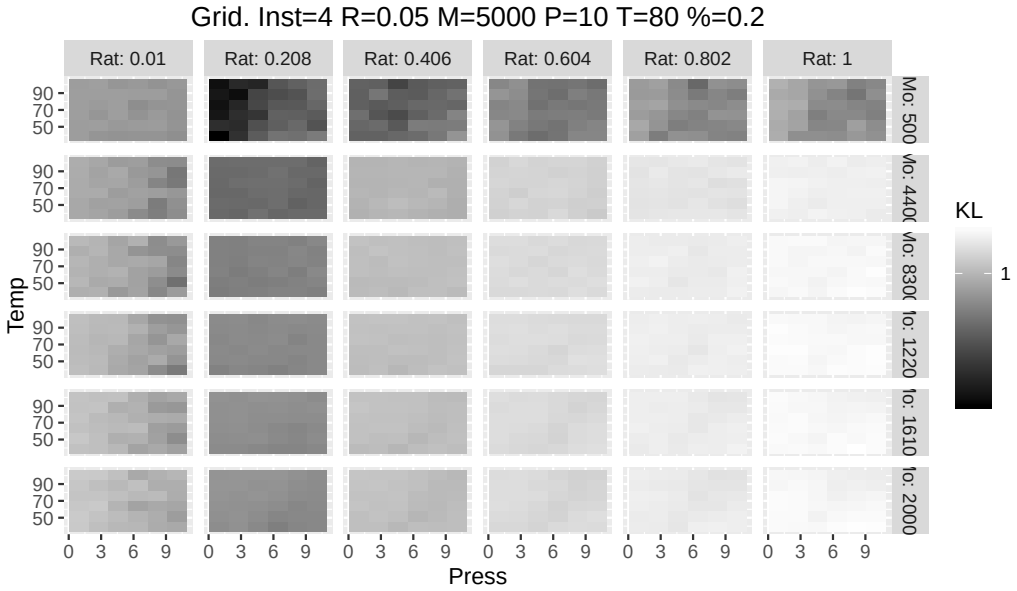


(b) 2 *Instantzia*

6.36. irudia: Soluzio Espazioaren Analisia 1, 2 Instantziak

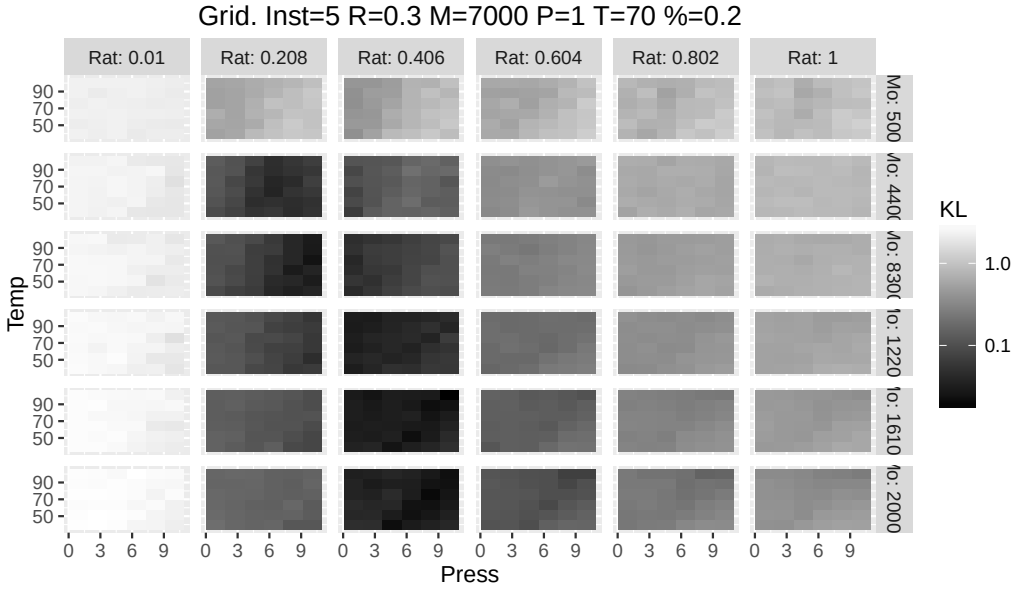


(a) 3 *Instantzia*

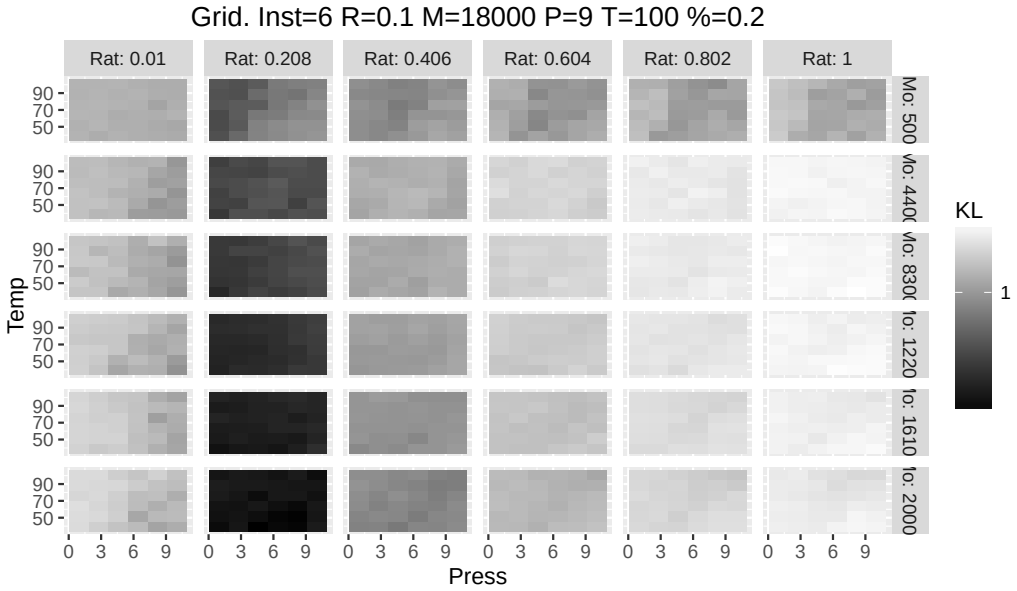


(b) 4 *Instantzia*

6.37. irudia: Soluzio Espazioaren Analisia 3, 4 Instantziak

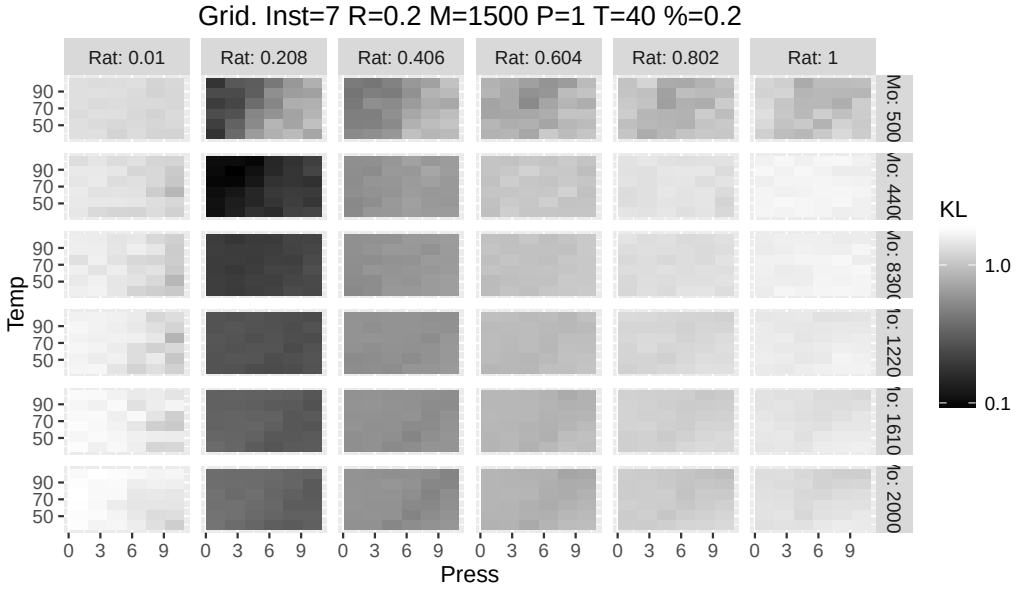


(a) 5 *Instantzia*

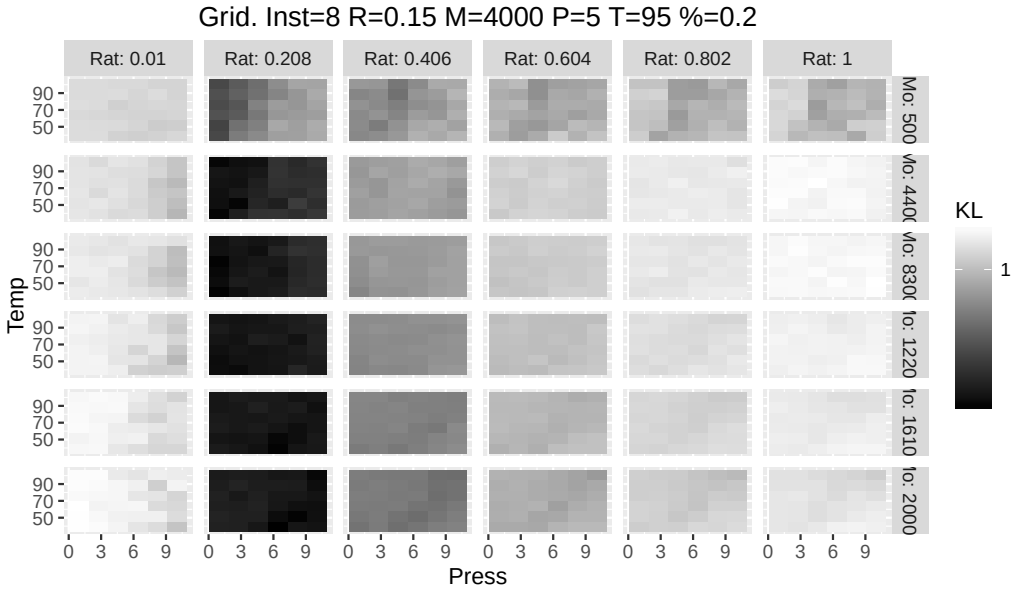


(b) 6 *Instantzia*

6.38. irudia: Soluzio Espazioaren Analisia 5, 6 Instantziak

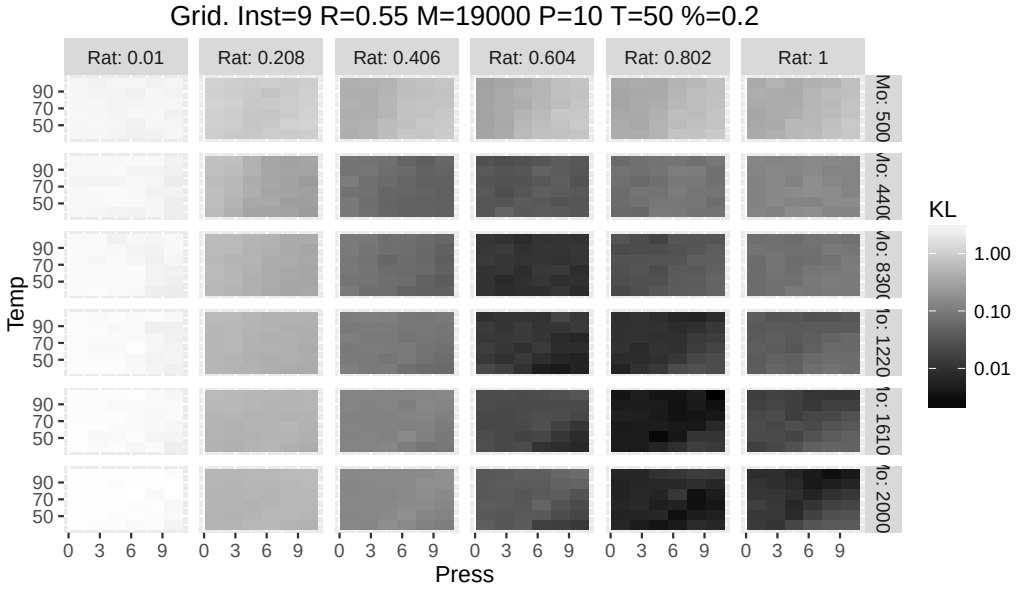


(a) 7 *Instantzia*

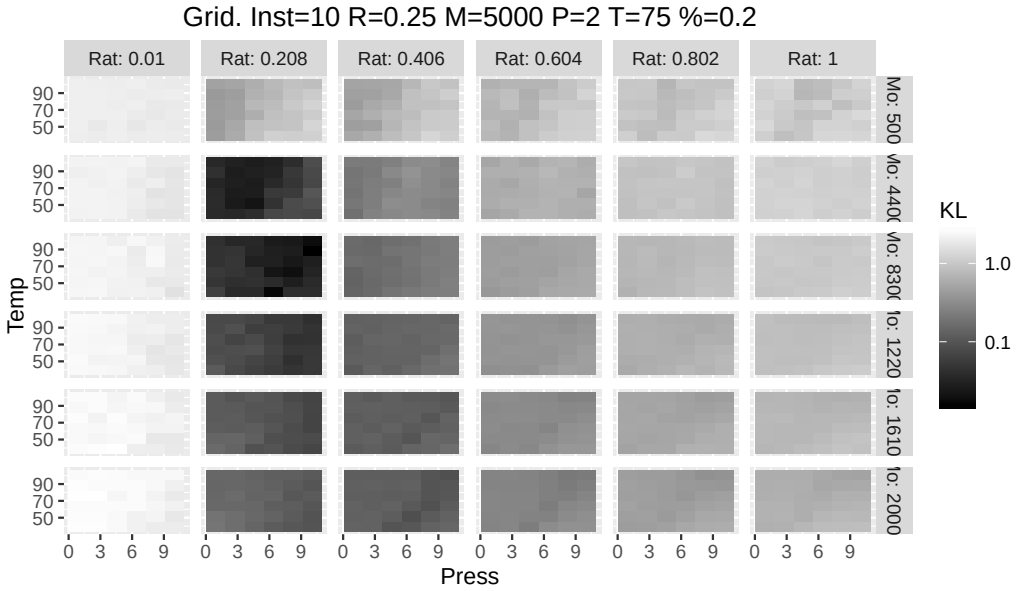


(b) 8 *Instantzia*

6.39. irudia: Soluzio Espazioaren Analisia 7, 8 Instantziak



(a) 9 *Instantzia*



(b) 10 *Instantzia*

6.40. irudia: Soluzio Espazioaren Analisia 9, 10 Instantziak