UZAMSAL KOŞUT VE DAĞITILMIŞ BENZETİMLER İÇİN İYİMSER BİR ZAMAN YÖNETİM MEKANİZMASI

AN OPTIMISTIC TIME MANAGEMENT MECHANISM FOR SPATIALLY PARALLEL AND DISTRIBUTED SIMULATIONS

BİLGE KAAN GÖRÜR

DOÇ. DR. KAYHAN İMRE
Tez Danışmanı

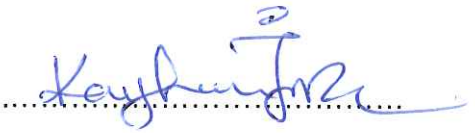
PROF. DR. HALİT OĞUZTÜZÜN
İkinci Tez Danışmanı

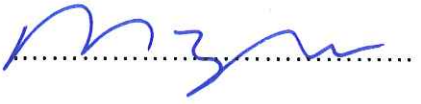
Hacettepe Üniversitesi
Lisansüstü Eğitim – Öğretim ve Sınav Yönetmeliğinin
Bilgisayar Mühendisliği Anabilim Dalı için Öngördüğü
DOKTORA TEZİ
olarak hazırlanmıştır.

2019

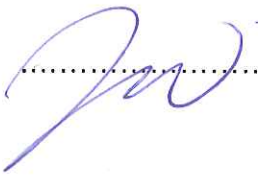
BİLGE KAAN GÖRÜR'ün hazırladığı "**Uzamsal Koşut ve Dağıtılmış Benzetimler için İyimser Bir Zaman Yönetim Mekanizması**" adlı bu çalışma aşağıdaki jüri tarafından **BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI**'nda **DOKTORA TEZİ** olarak kabul edilmiştir.

Prof. Dr. Özcan ÖZTÜRK
Başkan ..... 

Doç. Dr. Kayhan İMRE
Danışman ..... 

Doç. Dr. Murat MANGUOĞLU
Üye ..... 

Doç. Dr. Harun ARTUNER
Üye ..... 

Dr. Öğretim Üyesi Adnan ÖZSOY
Üye ..... 

Bu tez Hacettepe Üniversitesi Fen Bilimleri Enstitüsü tarafından **DOKTORA TEZİ** olarak ..... / ..... /..... tarihinde onaylanmıştır.

Prof. Dr. Menemşe GÜMÜŞDERELİOĞLU
Fen Bilimleri Enstitüsü Müdürü

ETİK

Hacettepe Üniversitesi Fen Bilimleri Enstitüsü, tez yazım kurallarına uygun olarak hazırladığım bu tez çalışmasında,

- tez içindeki bütün bilgi ve belgeleri akademik kurallar çerçevesinde elde ettiğimi,
- görsel, işitsel ve yazılı tüm bilgi ve sonuçları bilimsel ahlak kurallarına uygun olarak sunduğumu,
- başkalarının eserlerinden yararlanılması durumunda ilgili eserlere bilimsel normlara uygun olarak atıfta bulunduğumu,
- atıfta bulunduğum eserlerin tümünü kaynak olarak gösterdiğimi,
- kullanılan verilerde herhangi bir tahrifat yapmadığımı,
- ve bu tezin herhangi bir bölümünü bu üniversite veya başka bir üniversitede başka bir tez çalışması olarak sunmadığımı

beyan ederim.

16.04.2019

(imza)

BİLGE KAAN GÖRÜR

YAYIMLAMA VE FİKRİ MÜLKİYET HAKLARI BEYANI

Enstitü tarafından onaylanan lisansüstü tezimin/raporumun tamamını veya herhangi bir kısmını, basılı (kâğıt) ve elektronik formatta arşivleme ve aşağıda verilen koşullarla kullanıma açma iznini Hacettepe Üniversitesi'ne verdiğimi bildiririm. Bu izinle Üniversiteye verilen kullanım hakları dışındaki tüm fikri mülkiyet haklarım bende kalacak, tezimin tamamının ya da bir bölümünün gelecekteki çalışmalarda (makale, kitap, lisans ve patent vb.) kullanım hakları bana ait olacaktır.

Tezin kendi orijinal çalışmam olduğunu, başkalarının haklarını ihlal etmediğimi ve tezimin tek yetkili sahibi olduğumu beyan ve taahhüt ederim. Tezimde yer alan telif hakkı bulunan ve sahiplerinden yazılı izin alınarak kullanılması zorunlu metinleri yazılı izin alarak kullandığımı ve istenildiğinde suretlerini Üniversite'ye teslim etmeyi taahhüt ederim.

Yükseköğretim Kurulu tarafından yayınlanan ***"Lisansüstü Tezlerin Elektronik Ortamda Toplanması, Düzenlenmesi ve Erişime Açılmasına İlişkin Yönerge"*** kapsamında tezim aşağıda belirtilen koşullar haricince YÖK Ulusal Tez Merkezi / H.Ü. Kütüphaneleri Açık Erişim Sisteminde erişime açılır.

- ☐ Enstitü / Fakülte yönetim kurulu kararı ile tezimin erişime açılması mezuniyet tarihimden itibaren 2 yıl ertelenmiştir.
- ☐ Enstitü / Fakülte yönetim kurulu gerekçeli kararı ile tezimin erişime açılması mezuniyet tarihimden itibaren .... ay ertelenmiştir.
- ☐ Tezim ile ilgili gizlilik kararı verilmiştir.

16.04.2019

(imza)

BİLGE KAAN GÖRÜR

ÖZET

UZAMSAL KOŞUT VE DAĞITILMIŞ BENZETİMLER İÇİN İYİMSER BİR ZAMAN YÖNETİM MEKANİZMASI

Bilge Kaan GÖRÜR

Doktora, BİLGİSAYAR MÜHENDİSLİĞİ Bölümü
Tez Danışmanı: Doç. Dr. Kayhan Mustafa İMRE
Eş Danışman: Prof. Dr. Halit OĞUZTÜZÜN
Nisan 2019, 171 sayfa

Koşut ve dağıtılmış işlem teknolojileri, ağır işlem gücü gerektiren benzetimlerin kabul edilebilir sürelerde koşturulmasını sağlar. Benzetimlerin koşut veya dağıtılmış olarak çalıştırılmasındaki en önemli sorunlardan bir tanesi de koşut çalışan işlemler arasında yapılması gereken zaman uyumlamadır. Zaman uyumlamanın sağlıklı bir şekilde yapılmaması benzetim sonuçlarının beklenenden farklı, yanlış veya tutarsız olmasına neden olur. Koşut ve dağıtılmış benzetim sistemlerinde zaman uyumlama yöntemleri iki başlık altında ele alınmaktadır. Bunlardan ilki sakıngan zaman yönetim yaklaşımı olup, bu yaklaşımda koşut işlemlerin sürekli olarak tutarlı bir şekilde ilerlemesi sağlanır. Bunu yapabilmek için de koşut çalışan işlemler durum değişikliklerinden sonra bir zaman uyumlama sürecine girmek durumundadırlar. Bir diğer yaklaşım olan iyimser zaman yönetim yaklaşımı ise sakıngan zaman yönetim yaklaşımındaki zaman uyumlamadan kaynaklı gecikmeleri ortadan kaldırmayı amaçlar. Bunun için de zaman uyumlama mekanizmalarının gevşetilmesini önerir. İyimser zaman yönetim yaklaşımında koşut işlemler, zaman uyumlamayı beklemeden yerel benzetim zamanlarını ilerleterek diğer koşut işlemlerin daha ilerisinde bir zamana gidebilirler. Bu durumda, ilerleyen bir zamanda zaman

uyumlamanın gevşekliğinden kaynaklı sorunların ortaya çıkması mümkündür. Diğer koşut işlemlerin ilerisinde bir yerel zamana sahip bir koşut işlem, geçmiş zamanda işlemesi gereken bir iletiyi alabilir ve benzetimde tutarsızlıklar ortaya çıkar. Böyle bir durumda ilgili koşut işlem(ler)in geri sarma sürecine (benzetimin geçmiş bir zamanına geri dönerek ilgili sorunları çözmesi) girmesi söz konusudur.

Her iki zaman yönetim yaklaşımının birbirine göre artıları ve eksileri olduğu bilinse de, özellikle son 30 yılda iyimser yaklaşımların daha ölçeklenebilir ve tercih edilebilir olduğunu gösteren bir çok çalışmaya rastlanmaktadır. İyimser yaklaşımların başarımını daha da artırmanın bir yolu, maliyeti çok yüksek olabilen geri sarma süreçlerinin sayısını işlem bütünlüğünü bozmadan azaltabilmektir. Bu tez kapsamında geri sarma işlemlerini azaltarak iyimser yaklaşımların başarımını daha da artırmak için uzamsal-koşut benzetimlere yönelik genel amaçlı bir mekanizma önerilmiş ve bu mekanizmayı kullanan bir uygulama çatısı geliştirilmiştir. Mevcut yöntemlerde geri sarma işlemlerinin tamamen gereksiz olduğu bazı durumlarda bile gerçekleştirilmesi mümkün olabilmektedir. Önerilen mekanizmanın asıl amacı bu gereksiz geri sarma işlemlerinin tespit edilmesi ve bunlardan kaçınılabilmesinin sağlanmasıdır. Benzetimde henüz bir tutarsızlık ortaya çıkmadan önce, ileri tarihte ortaya çıkabilecek geri sarmaların hangi durumlarda gerçekten gerekli olduğu, hangi durumlarda ise kaçınılabilir olduğu önerilen yöntem sayesinde belirlenir. Bu özelliğinden dolayı önerilen yönteme Öngörülmüş Geri Sarma mekanizması adı verilmiştir. Yapılan deneyler Öngörülmüş Geri Sarma mekanizmasının bilinen iyimser yaklaşımlara göre kayda değer hızlanmalar sağladığını göstermiştir. Ayrıca bu ana katkısının yanı sıra tez çalışmaları kapsamında; bir etmen tabanlı modelleme ve benzetim yazılımı iyimser zaman yönetimi yaklaşımıyla genişletilmiş, denetim noktası alma zamanlarını belirleyen dinamik yaklaşımlar için yeni bir yöntem uygulanmış ve geri sarma sayısının durum kaydetme mekanizmalarını iyileştiren bir yöntemle azaltılabileceği de gösterilmiştir.

Anahtar Kelimeler: Koşut ve dağıtılmış benzetim, Zaman uyumlama, İyimser zaman yönetimi, Etmen tabanlı modelleme ve benzetim, Uzamsal-koşut benzetim.

ABSTRACT

AN OPTIMISTIC TIME MANAGEMENT MECHANISM FOR SPATIALLY PARALLEL AND DISTRIBUTED SIMULATIONS

Bilge Kaan GÖRÜR

Doctor of Philosophy, Department of Computer Engineering
Supervisor: Assoc. Prof. Dr. Kayhan Mustafa İMRE
Co-Supervisor: Prof. Dr. Halit OĞUZTÜZÜN
April 2019, 171 pages

Parallel and distributed simulation enables to run simulations that have heavy computational loads within acceptable time periods. One of the most important problems during the execution of parallel and distributed simulation is the synchronization of parallel processes. To make an unreliable synchronization causes simulation results to be unexpected, wrong or inconsistent. Synchronization methods that are used in parallel and distributed simulations are classified into two main groups. The first one is called conservative time management approach and continuously tries to make parallel processes consistent. Therefore, parallel processes should be synchronized after state transitions. Objective of the other approach, namely optimistic time management, is getting rid of latencies originating from too many synchronizations. Therefore, optimistic time management allows to loosen synchronization mechanism. In an optimistic time management approach, parallel processes can advance to a time ahead of other parallel processes by advancing their local time without caring about synchronization. It is possible that some issues originating from loosened synchronization can show up in a further simulation time. A parallel process which has a greater simulation time than others

may receive a message with a past timestamp from them and some inconsistencies may appear in the simulation. In this case, the relevant parallel processes have to recover the simulation state by performing a rollback operation that goes back in the simulation and figures out the issues.

Both time management approaches have their own pros and cons. However, in the last three decades, a lot of studies in the literature that show optimistic time management can be more scalable and preferable appeared. One of the things that can improve the performance of optimistic approaches is reducing the number of rollbacks without breaching integrity of the model. In this thesis, a generic mechanism that improves performance of optimistic time management approaches by reducing number of rollbacks has been proposed for spatially-parallel agent-based simulations. A framework that uses this mechanism has been developed. In the existing methods, a lot of rollback operations may be performed even if they are not necessary. The objective of the proposed method is detecting these unnecessary rollbacks and avoiding to perform them. For this purpose, much before an inconsistency is observed, the proposed method determines if a rollback operation should be performed or not for the past time-stamped messages that will be received in the future. Because of this feature, the proposed method is called Predetermined Rollbacks. The experiments showed that Predetermined Rollbacks achieved significant speedup against conventional optimistic time management mechanisms. In addition to the main contribution, an agent-based modeling and simulation software has been extended with optimistic time management; a well-known control algorithm that is widely used in computer networks has been employed as a dynamic checkpointing scheduler. It has been shown that the number of rollbacks can be reduced by a method that improves the state saving mechanisms.

Keywords: Parallel and distributed simulation, Synchronization, Optimistic time management, Agent-based modeling and simulation, Spatially parallel simulation.

TEŞEKKÜR

Tüm hayatım boyunca her an yanımda olan ve hiçbir zaman desteklerini esirgemeyen anneme, akademik çalışmalar konusunda tavsiyelerine ve teşviklerine her an ihtiyaç duyduğum babama ve abime;

Lisansüstü eğitimim boyunca vermiş olduğu tüm emek ve destekleri ile paylaştıkları çok değerli bilgiler için danışmanlarım Doç Dr. Kayhan Mustafa İMRE ve Prof. Dr. Halit OĞUZTÜZÜN'e;

Tez çalışması boyunca fikirleriyle tezin gelişimine verdiği değerli katkılardan dolayı Prof. Dr. Levent YILMAZ'a; tez izleme komiteleri boyunca çalışmalarımı takip ederek değerli önerilerde bulunan Dr. Deniz ÇETİNKAYA'ya, Doç Dr. Murat MANGUOĞLU'na ve Dr. Adnan ÖZSOY'a;

Tez çalışması boyunca yapmış olduğum deneyler için ihtiyacım olan hesaplama ortamlarını sağlayan ODTÜ-TSK MODSİMMER'e, ODTÜ Bilgisayar Mühendisliği Bölümü'ne ve TÜBİTAK'a;

Son olarak da tez çalışmam boyunca sürekli olarak beni çalışmaya teşvik eden, en sıkıntılı zamanlarımda yanımda olan ve bana katlanan biricik eşim Betül ÖZKAN GÖRÜR'e

Tüm içtenliğimle teşekkür ederim.

Nisan 2019, Ankara

İÇİNDEKİLER

ÇİZELGELER

ŞEKİLLER

SİMGELER VE KISALTMALAR

Simgeler

α	:	TAÇA algoritmasında kullanılan artış sabiti
β	:	TAÇA algoritmasında kullanılan azalış çarpanı
σ	:	Standart sapma
C_{AM}	:	Karşıt iletilerin hazırlanması ve gönderilmesinin maliyeti
C_{CF}	:	İleri sarma işleminin maliyeti
C_R	:	Geri sarma işleminin toplam maliyeti
C_{RE}	:	Benzetim adımlarının yeniden oynatılmasının maliyeti
C_{SD}	:	Durum farkı hesaplama yönteminin maliyeti
C_{SMI}	:	Bir mızıkçı iletinin incelenmesinin maliyeti
C_{SR}	:	Benzetim durumunun bir önceki denetim noktasına çekilmesinin maliyeti
C_{SS}	:	Benzetim durumunun kaydedilmesinin maliyeti
E	:	Verimlilik değeri
p	:	Koşut bir benzetimin çalıştırıldığı işlemci çekirdeği sayısı
S	:	Hızlanma değeri
T_S	:	Benzetimin tek bir mantıksal işlem tarafından çalıştırılması durumunda geçen zaman
T_P	:	Benzetimin çok sayıda mantıksal işlem tarafından koşut olarak çalıştırılması durumunda geçen zaman
χ_t	:	Denetim noktası alma sıklığı

Kısaltmalar

ETMB	:	Etmen tabanlı modelleme ve benzetim
GCC	:	GNU Compiler Collection
GPU	:	Graphics Processing Unit (Grafik İşlemci Birimi)

KB	:	Kilobayt
Mİ	:	Mantıksal işlem
MPI	:	Message Passing Interface (İleti Gönderim Arayüzü)
ÖGS	:	Öngörülmüş Geri Sarma
Repast HPC	:	Repast for High Performance Computing
RHPC_PR	:	Öngörülmüş Geri Sarma mekanizmasını kullanan Repast HPC sürümü
RHPC_SS	:	Alt durum kaydetme mekanizmasını kullanan Repast HPC sürümü
RHPC_TW	:	Zaman Bükülmesi mekanizmasını kullanan Repast HPC sürümü
RMI	:	Remote Method Invocation (Uzak Yordam Başlatma)
ODTÜ	:	Orta Doğu Teknik Üniversitesi
ODTÜ-TSK MODSİMMER	:	Orta Doğu Teknik Üniversitesi – Türk Silahlı Kuvvetleri Modelleme ve Simülasyon Araştırma ve Uygulama Merkezi
TAÇA	:	Toplamsal Artış Çarpımsal Azalış
TRUBA	:	Türk Ulusal Bilim e-Altyapısı
TÜBİTAK	:	Türkiye Bilimsel ve Teknolojik Araştırma Kurumu
U/D	:	Uygulanabilir değil
YBH	:	Yüksek Başarımlı Hesaplama

1. GİRİŞ

Benzetimlerin koşut ve dağıtılmış olarak çalıştırılması bilgisayarlı ortamda benzetim araştırmalarının başlamasından bu yana çalışılmakta olan konulardan bir tanesidir. Teknoloji geliştikçe benzetimlerden beklentiler artmakta ve araştırmacılar daha yüksek sadakat seviyeli ve daha gerçekçi benzetimler elde etmeyi amaçlamaktadır. Eldeki mevcut kaynaklardan azami seviyede yararlanılması da bu amaca ulaşmakta önemli bir adım oluşturmaktadır. Koşut benzetimlere başvurulmasının nedeni daha fazla bilgisayar kaynağının koşut kullanılmasıyla benzetimlerin hızlı bir şekilde tamamlanmasının istenmesidir [1]. Bir benzetimin, bir bilgisayar üzerindeki çok sayıda işlemci çekirdeği veya işlemci tarafından; veya ortak bir ağ üzerindeki çok sayıda bilgisayar tarafından çalıştırılmasıyla önemli hızlanmalar elde edilebilir. Böylece sadakat seviyesi yüksek benzetimlerin daha kısa sürede tamamlanması sağlanabilmektedir. Örneğin; [2] çalışmasında bir benzetimin bir yüksek başarımlı hesaplama (YBH) sisteminde koşut çalıştırılarak sonuçların çok daha hızlı alındığı gözlenmiştir. [3] çalışmasında dünya benzetim zamanı rekoru kıran bir koşut uygulama örneği görülebilir. Benzer şekilde benzetimlerden daha doğru sonuçlar elde edebilmek amacıyla da milyonlarca koşum alınması gerekebilmektedir. Monte Carlo benzetimleri bu durumlar için bilinen bir örnektir. Monte Carlo benzetimlerinin koşut çalıştırılmasıyla istenen doğru sonuçlara daha hızlı bir şekilde ulaşmak da yine koşut benzetim tekniklerinin kullanılmasıyla gerçekleşebilir [4]. Dağıtılmış benzetimlere başvurulmasının temel nedeni ise doğası gereği birbirinden bağımsız çalışabilen benzetimlerin birlikte çalışabilirliğinin sağlanmasıdır. Böylece birbirinden bağımsız geliştirilmiş benzetim uygulamalarının aynı ortamda çalıştırılması mümkün olabilmektedir. Benzetimlerin birlikte çalışabilirliği ve yeniden kullanılabilirliği için uzun yıllardır çeşitli standartlar geliştirilmiş olup, birçok farklı uygulamada bu standartlardan yararlanılmıştır. High Level Architecture (HLA) [5] ve Distributed Interactive Simulations (DIS) [6] bu standartlar arasından en bilinen iki tanesidir. [7] ve [8] kaynaklarındaki çalışmalar HLA kullanılarak geliştirilmiş ve kıtalararası çalıştırılmış bir benzetim sistemi örneğidir.

Modelleme ve benzetim çalışmalarının önemli bir kısmını da etmen tabanlı modelleme ve benzetim (ETMB) yaklaşımı oluşturmaktadır. Literatürde bir çok

kaynakta farklı etmen tanımlarına rastlamak mümkündür. Bu tanımlar göz önünde bulundurularak etmen şu şekilde özetlenebilir: “otonom bir davranışı olan, bulunduğu ortamı ve diğer etmenleri algılayarak eylemini gerçekleştiren ve ortama etkide bulunan yazılım varlıkları” [9][10]. ETMB, karmaşık problemleri çözmek için basit yaklaşımlar önerir. İyi tanımlanmış etmenler problem tanımını ve modellemeyi daha kolay bir hâle indirger [11]. Bilgisayar bilimlerinin yanı sıra biyoloji, sosyoloji, ekonomi ve benzeri birçok disiplinde de ETMB yaklaşımından yararlanılır. İşlem gücü teknolojisindeki gelişmeler gün geçtikçe ETMB yaklaşımını daha da bilinir hâle getirmektedir [11]. Ayrıca doğası gereği bazı problemlerin ETMB yaklaşımıyla modellenmesi yazılım geliştiricilerin işini kolaylaştırır [11][12].

Gerçekçiliği yüksek etmen tabanlı benzetimlerde çok yüksek sayıda etmen kullanılması gerekebilmektedir. Bu benzetimlerin hızlı bir şekilde tamamlanabilmesi için koşut benzetim teknolojilerinden yararlanmak mümkündür. YBH sistemlerinde koşut ve dağıtılmış çok etmenli benzetimlerin geliştirilmesine yönelik geniş bir literatür taraması [13] kaynağında görülebilir. Etmen tabanlı modellerin koşutlaştırılması iki şekilde gerçekleştirilebilir. Bunlardan ilki etmen-koşut, ikincisi ise uzamsal-koşut (ortam-koşut) olarak adlandırılabilir [14]. Etmen-koşut yaklaşımda benzetimdeki etmenler işlemciler arasında paylaştırılır. Böylece her bir işlemci, etmenlerin bir alt kümesini yönetmekten sorumludur. Uzamsal-koşut yaklaşımda ise her bir işlemci etmenlerin yaşadığı benzetim uzayının belirli bir kısmını yönetmekten sorumludur. Her iki yaklaşımın da olumlu ve olumsuz yönleri bulunmaktadır. Etmen-koşut yaklaşımda, etmenler arası etkileşimin uzaklık tabanlı yapılması, bir başka deyişle komşuluk ilişkisi üzerinden yapılması, işlemciler arasında çok fazla iletişim gerektireceği için benzetimin çok yavaşlamasına neden olabilir. Uzamsal-koşut yaklaşımın en belirgin olumsuzluğu ise etmenlerin uzayın belli bir kısmında kümelenmesi durumunda ortaya çıkar. Etmenlerin büyük çoğunluğu bir kısım işlemci tarafından yönetilirken diğer işlemciler çok daha az etmeni yönetmek durumunda kalabilir. Bu durumda işlemcilerden bir kısmı diğerlerinden çok daha hızlı çalışacak ve diğer işlemcilerin önüne geçecektir. Dolayısıyla işlemciler arasındaki koşutluk kaybolacak ve benzetim ardıl bir şekilde çalışmaya başlayacaktır [15]. Bu durumda işlemcilerin yönettiği uzayın dinamik olarak değişmesiyle yük dengelemesi (İng. *load balancing*) yapılarak koşutluğun faydalarından yararlanılabilir.

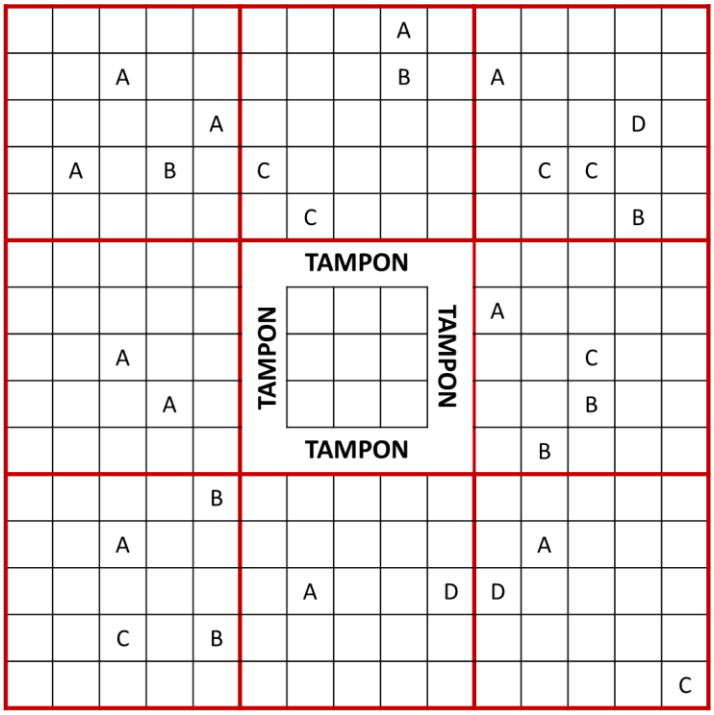
Bu tez çalışmasında birçok etmen tabanlı model için kullanılan uzamsal-koşut benzetimler hedef alınmıştır. Bunun en önemli nedenlerinden bir tanesi uzamsal-koşut benzetimlerin, bağımsız çekirdekleri bir torus yapısında yerleştirilmiş işlemci mimarilerine kolay bir şekilde eşleştirilebilecek olmasıdır. Bu işlemci mimarileri bugün için deneysel olarak geliştirilmiş olsa da önümüzdeki yıllarda koşut işlem teknolojilerinin etkinliğini artırmaya yönelik ümit vaad etmektedir. Chen ve çalışma arkadaşları tarafından önerilen işlemci mimarisi [16] ve KiloCore mimarisi [17] bu mimarilere bir örnektir. Öte yandan, bir çok farklı disiplinde ortaya çıkan model örneklerinin uzamsal koşut olarak çalıştırılması mümkündür. Örneğin; hücresel özdevinirlerin [18], biyoloji ve sağlık alanında geliştirilen bazı modellerin [19][20], moleküler benzetimlerin [21] ve trafik benzetimlerinin [22] koşut çalıştırılması bu örneklerden sadece bir kaçıdır. Ayrıca uzamsal koşut benzetimler için, bu tez çalışmasında da hedef alınan iyimser zaman yönetim mekanizmalarını kullanarak iyileştirmeler sağlayan çalışmalara da literatürde rastlanmaktadır [23][24][25].

Şekil 1.1'de 2 boyutlu bir ızgara uzayın 9 işlemci tarafından nasıl paylaşıldığı gösterilmektedir. Bu örnekte her bir işlemci 5×5'lik bir alt uzayı yönetmektedir. Etmenler arasında bir etkileşim ortaya çıktığı zaman, o etmeni yöneten işlemci gerekli etkileşimi yerel olarak ele alır. Eğer etmenler alt uzayın sınırlarında yer alıyorsa, bu etmen komşu işlemcide yer alan etmenlerle etkileşime girebilecek uzaklıkta olabilirler. Bu durumda sınıra yakın etmenler komşu işlemciye bildirilir ve komşu işlemcinin de bu etmeni kendi hesaplamalarında kullanılması sağlanır. Doğru bir hesaplama yapılabilmesi için işlemciler arasındaki bu etkileşimin doğru zamanda yapılması gerekmektedir. Bu tez çalışmasında kullanılan ETMB yazılımı Repast HPC terminolojisinde bu sınır bölgeler tampon alan (İng. *buffer*) olarak adlandırılmıştır [26]. Dolayısıyla bu çalışmanın kalan kısmında sınır alanları tampon alan olarak anılacaktır.

Bu tez kapsamında, kaynakların koşut kullanımıyla etmen tabanlı benzetimlerin hızlandırılmasına yönelik literatürde bulunan çözümlerin iyileştirilmesi sağlanmıştır. Sunulan uygulama çatısının literatüre katkıları şu şekildedir:

- Etmen tabanlı ve koşut çalışabilen bir modelleme ve benzetim yazılımı sunulmuştur.

- Bilgisayar ağlarında kullanıldığı bilinen “Toplamsal Artış Çarpımsal Azalış” (TAÇA) [27] algoritması, zaman yönetim mekanizmalarına uyarlanarak bu algoritmanın denetim noktası sıklığını belirlemede kullanılması sağlanmıştır.
- Özgün bir alt-durum kaydetme mekanizması sunulmuştur.
- Özgün olarak geliştirilen “Öngörülmüş Geri Sarma mekanizması” ile, gereksiz geri sarmaların önüne geçilmiş; böylece benzetimlerin daha hızlı çalışması sağlanmıştır.



Şekil 1.1. 9 koşut işlem tarafından yönetilen iki boyutlu bir ızgara ortamı

Bu dokümanın kalan kısmı şu şekilde yapılandırılmıştır:

- Bölüm 2’de, bu tez çalışmasının temelini oluşturan zaman yönetim mekanizmalarının çalışma ilkeleri ayrıntılarıyla anlatılmış olup, bu bölüm temel bir bilgi niteliğindedir.
- Bölüm 3’te, yine bu tez çalışması kapsamında sıklıkla başvurulan denetim noktası mekanizmaları hakkında temel bilgiler verilmiştir.
- Bölüm 4, literatürde yer alan ve bu tez çalışması ile ilgili olan çalışmaların kısa bir özetini içermektedir.
- Tez çalışması kapsamında yapılan denemelerde açık kaynak kodlu Repast HPC yazılımı genişletilerek kullanılmıştır. Bölüm 5’te neden Repast HPC

yazılımının seçildiği ve bu yazılımın Zaman Bükülmesi mekanizması ile nasıl genişletildiği anlatılmıştır.

- Bölüm 6'de Zaman Bükülmesi mekanizmasının iyileştirilmesine yönelik olarak tez çalışması sırasında ortaya konan özgün alternatif yöntemlerden bahsedilmiştir. Bu çalışmalardan Alt-Durum Kaydetme mekanizmasının başarımı Zaman Bükülmesi mekanizmasıyla karşılaştırılmıştır
- Bölüm 7'de özgün olarak geliştirilen Öngörülmüş Geri Sarma mekanizması ayrıntılarıyla anlatılmış ve başarımı Zaman Bükülmesi mekanizmasıyla karşılaştırılmıştır.
- Son bölüm olan Bölüm 8'de ise bu tez çalışması sonucunda ortaya konan çalışmaların değerlendirmesi yapılmış, literatüre olan katkısı incelenmiş ve gelecek çalışmalara yönelik öneriler sunulmuştur.

Bu tez çalışmasında sıkça kullanılan terimlerin Türkçe – İngilizce karşılıkları ve tanımları sırasıyla Ek - 1 ve Ek - 2'de verilmiştir.

2. KOŞUT VE DAĞITILMIŞ BENZETİMLERDE ZAMAN YÖNETİM MEKANİZMALARI

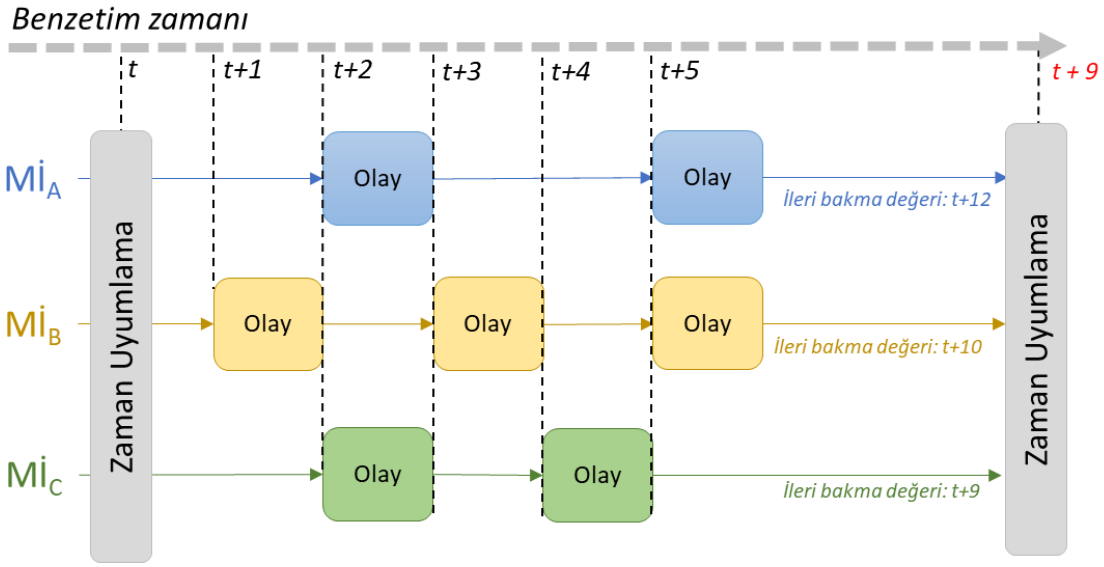
Koşut ve dağıtılmış benzetimlerde yaygın olarak çalışılan konulardan bir tanesi koşut işlemlerin zaman uyumunun sağlanmasıdır. Bu amaçla 40 yıldan fazla bir süredir zaman yönetim mekanizmaları çalışılmaktadır. Bu mekanizmalar, benzetimlerin hızlandırılmasında önemli bir rol oynayabilmektedir. Zaman yönetim mekanizmalarına yönelik literatür özeti ve sorunların önemli bir kısmı [1], [28] ve [29] kaynaklarında bulunabilir. Zaman yönetim mekanizmaları iki ana başlık altında ele alınmaktadır. Bunlardan bir tanesi sakıngan yaklaşım, bir diğeri ise iyimser yaklaşım olup, yöntemlerin ayrıntıları aşağıda verilmiştir.

Koşut ve dağıtılmış benzetim literatüründe koşut olarak çalışan her bir görev genellikle mantıksal işlem (Mİ) olarak anılmaktadır. Bu nedenle tezin kalan kısmında koşut olarak çalışan görevler mantıksal işlem adıyla anılacaktır.

2.1. Sakıngan Zaman Yönetim Mekanizması

Sakıngan yaklaşımı kullanan benzetim sistemlerindeki Mİ'ler benzetim zamanlarını zaman uyumlu bir şekilde ilerletirler. Bir Mİ, ancak diğer Mİ'lerden artık kendisine bir olay atanmayacağını garantilediğinde yeni bir olayı çalıştırmaya başlar. Bir başka deyişle sakıngan yaklaşım, bir Mİ'ye geçmiş zamanlı bir olay atanmasını engeller. Sakıngan yaklaşım, bu nedenle Mİ'lerin eylemlerinden sonra zaman uyumlanmasını sağlar. Sakıngan yaklaşımda, bir Mİ'nin ne kadar zaman boyunca diğer Mİ'lere en az ne kadar zaman boyunca bir olay atamayacağını gösteren değişkene "ileri bakma" (İng. *lookahead*) adı verilir. Bazı benzetimler ileri bakma değerini sıfır olarak da kullanabilir (İng. *zero-lookahead*). Ancak ileri bakma değerinin küçük tutulduğu benzetimlerde, sakıngan yaklaşımların iyi bir başarım elde edemediği bilinmektedir [30]. Sakıngan yöntemlerin temel çalışma ilkelerini açıklayan bir örnek Şekil 2.1'de verilmiştir. Bu örnekte üç Mİ'ye ait olayların işlenişi gösterilmiştir. t zamanındaki bir zaman uyumlamadan sonra her bir Mİ, yerel olaylarını kendi içerisinde gerçekleştirir. Bu olaylar işlendikten sonra tüm Mİ'ler en az hangi benzetim zamanına kadar diğer Mİ'lere olay atamayacaklarını birbirlerine bildirirler. Bu bildirilen benzetim zamanları arasından en küçük olan benzetim zamanında bir zaman uyumlama gerçekleştirilir.

Bu zaman uyumlama sayesinde Mİ'lerin birbirinden geçmiş zaman damgalı bir olay almaması sağlanmış olur.



Şekil 2.1. Sakıngan zaman yöntemlerinin çalışma ilkesini gösteren bir örnek

Gerçekleştiriminin basit olması ve güvenilir bir yöntem sağlaması dolayısıyla sakıngan yöntemler bir çok koşut ve dağıtılmış benzetim uygulamasında yaygın bir şekilde kullanılmaktadır. Ancak sürekli olarak Mİ'lerin zaman uyumlanmasının gerekmesi, benzetimdeki koşutluğun kaybedilmesine ve benzetimin daha geç sürelerde tanımlanmasına neden olabilmektedir [1].

2.2. İyimser Zaman Yönetim Mekanizması

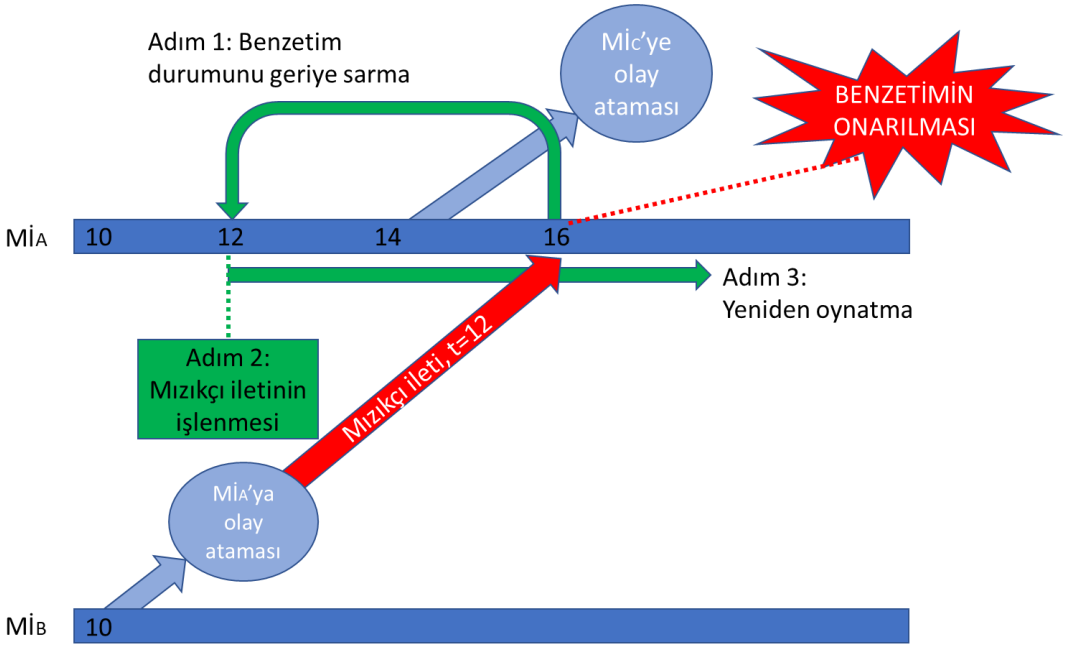
Sakıngan yaklaşımlara alternatif olarak iyimser zaman yönetim mekanizmaları da koşut ve dağıtılmış benzetimlerde kullanılan yöntemlerdendir. İyimser yaklaşımlarda bir Mİ, diğer Mİ'lerin benzetim zamanını bilmeden kendi sorumluluğundaki olayları işletir. Sakıngan yaklaşımlardan en önemli farkları zaman uyumlama noktalarıdır. İyimser yaklaşımda Mİ'lerin sürekli olarak birbirleriyle zaman uyumlanması gerekmez; bunun yerine zaman uyumlama belirli aralıklarla yapılabilir. Gevşetilmiş zaman uyumlama, Mİ'ler arasındaki tutarlılığın bozulmasına ve nedensellik hatalarına yol açabilir. Çünkü iyimser Mİ'lerin yerel zamanı birbirlerinden bağımsız bir şekilde ilerlemektedir ve bir Mİ kendi yerel zamanından daha eski bir ileti alabilir. Bu tür iletiler, bu çalışmanın kalan kısmında mızıkçı ileti (İng. *straggler message*) olarak adlandırılacaktır. Mızıkçı ileti alan bir Mİ, ortaya çıkacak nedensellik hatalarını

onarmak zorundadır. Bunun için Mİ, zamanda bir geri sarma işlemi başlatır. İyimser yaklaşımlar nedensellik hatalarına açık olduğu için özel bir dikkat gerektirir [1]. İyimser yaklaşımlar gevşetilen zaman uyumlamadan kaynaklı bir başarım kazancı sağlar; ancak bazı benzetimlerde geri sarma işlemleri çok maliyetli olabilmekte ve bu kazanç kaybolmaktadır.

İyimser yaklaşımların en bilinen örneği Jefferson tarafından önerilen Zaman Bükülmesi mekanizması [31] olup halen geliştirmeye açık yönleri bulunmaktadır. Bu tezde yapılan çalışmalar kapsamında da Zaman Bükülmesi mekanizması kullanılmıştır. Zaman Bükülmesi mekanizmasının temel çalışma ilkelerini açıklayan bir örnek Şekil 2.2'de verilmiştir. Bu örnekte $Mİ_B$, yerel benzetim zamanı 10 iken, $Mİ_A$'ya 12. benzetim zamanında işlenmek üzere bir olay atamaktadır. Ancak Mİ'ler arasında bir zaman uyumlama olmadığı için $Mİ_A$, bu iletiyi yerel zamanı 16 iken almaktadır. Dolayısıyla $Mİ_A$, benzetim zamanını 12'ye geri sararak benzetimi onarmak durumunda kalır (bu geri sarma işlemi tersine hesaplama veya Bölüm 3'te ayrıntıları verilen denetim noktaları yöntemleri kullanılarak yapılabilir). Benzetim zamanı geriye sarıldıktan sonra, $Mİ_B$'den gelen geçmiş zaman damgalı ileti de hesaba katılarak benzetimin doğru duruma gelmesi sağlanır. Ardından $Mİ_A$ 12. benzetim zamanından başlayarak işlemlerine devam eder. Bu sırada $Mİ_A$'nın dikkat etmesi gereken noktalardan bir tanesi de geri sarma işleminden önce başka bir Mİ'ye bir olay atayıp atamadığıdır. Şekil 2.2'deki örnekte $Mİ_A$'nın $Mİ_C$'ye 14. benzetim zamanında bir olay atadığı görülmektedir. Yeniden oynatma aşamasında bu iletinin hatalı olduğu veya gereksiz yere gönderildiği $Mİ_A$ tarafından fark edilirse, $Mİ_C$'nin bu iletinin dikkate alınmaması konusunda uyarılması gerekir. Bu amaçla $Mİ_A$ bir karşıt ileti oluşturarak $Mİ_C$'yi bilgilendirir. Karşıt iletiyi alan $Mİ_C$, eğer daha önceden alınan iletiyi henüz işlememişse hem karşıt iletiyi hem de hatalı iletiyi olay kuyruğundan silerek dikkate almaz. Ancak hatalı ileti zaten işletilmişse, $Mİ_C$ de bir benzetim onarım işlemi başlatmalıdır [1].

İyimser zaman yönetim yaklaşımları genel olarak Zaman Bükülmesi mekanizması etrafında toplanmış olup, bu mekanizmaya eklenti olarak geliştirilmiş bir çok yöntem bulunmaktadır. Tembel iptal etme (lazy cancellation), fosil toplama (fossil collection) ve tembel yeniden değerlendirme (lazy re-evaluation) bu yöntemlerden en

bilinenleridir. Bu mekanizmalar tez kapsamında ele alınmadığı için ayrıntılı olarak bahsedilmemiş olup, daha kapsamlı bilgi [1], [32] ve [33] kaynaklarında bulunabilir.



Şekil 2.2. Zaman Bükülmesi mekanizmasının çalışma ilkesini gösteren bir örnek

2.3. Sakıngan ve İyimser Zaman Yönetim Mekanizmalarının Karşılaştırılması

Sakıngan ve iyimser yaklaşımların nitelikleri arasındaki belirgin farklar Çizelge 2.1'de verilmiştir. Her iki yaklaşımın olumlu ve olumsuz yönlerini sunan bazı çalışmalar [28], [32] ve [34] kaynaklarında görülebilir. [35] kaynağında ise çok büyük ölçekte koşut (İng. *massively parallel*) modeller için hangi zaman yönetim mekanizmasının seçilmesinin daha iyi olacağına yönelik kılavuz bir çalışma sunulmuştur. [36] çalışmasında da iyimser ve sakıngan zaman yönetim mekanizmalarının karşılaştırılması amacıyla gerçekleştirilmiş bir durum çalışması görülebilir. Başarım açısından bakıldığında sakıngan veya iyimser yaklaşımlardan birinin diğerine üstün geldiğini söylemek mümkün değildir. Fakat; özenli bir şekilde tanımlanmış iyimser yaklaşımları kullanarak benzetimlerin tamamlanma sürelerini kısaltan birçok çalışmaya literatürde rastlamak mümkündür. Bunun temel nedeni ise benzetimin gereksiz zaman uyumlama işlemlerinden kurtarılmasıdır. Örneğin Presley ve çalışma arkadaşları Sharks World örneğini Zaman Bükülmesi ve geç iptal etme mekanizmasıyla birlikte kullanarak %29,5'lik bir hızlanma sağlamışlardır [37]. [2] ve [38] çalışmalarında ise IBM Blue Gene gibi modern YBH platformları

kullanılarak Zaman Bükülmesi mekanizmasının ölçeklenebilirliği gösterilmiştir. Örneğin; Barnes ve çalışma arkadaşları 2013 yılında Zaman Bükülmesi mekanizmasını Sequoia IBM Blue Gene/Q süper bilgisayarında kullanarak benzetim hız rekorunu kırmışlardır [3].

Çizelge 2.1. Sakıngan ve İyimser Zaman Yönetim Mekanizmalarının Karşılaştırılması

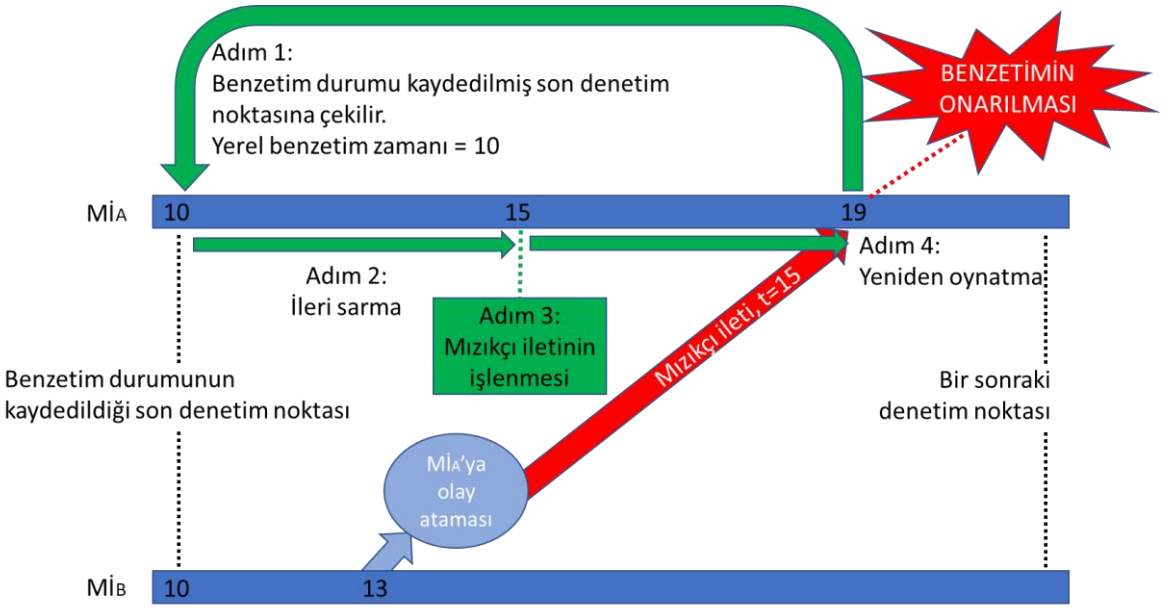
Sakıngan Zaman Yönetim Yaklaşımı	**İyimser Zaman Yönetim Yaklaşımı**
Sıkı bir zaman uyumlama söz konusudur.	Gevşek bir zaman uyumlama söz konusudur.
Olayların güvenli bir şekilde işletimi ön plandadır.	Olaylar güvenli bir şekilde işletilmediği için özel bir dikkat gerektirir.
Gerçekleştirimi basittir.	Denetim noktaları, geri sarma ve ileri sarma gibi gelişmiş mekanizmalar gerektirir.
Ağdaki maliyetlerden kaynaklı olarak pahalı olabilir.	Koşut işlemler arasındaki etkileşimin yüksek olması durumunda ortaya çıkan çok sayıdaki geri sarma işleminden dolayı pahalı olabilir.

Yukarıdaki örneklere bakıldığında Zaman Bükülmesi mekanizmasının daha iyi sonuçlar verebildiği görülse de zaman uyumlamanın sık gerekeceği benzetimlerde iyimser bir zaman yönetim mekanizması kullanmak benzetimin yavaşlamasına da neden olabilir. Bunun nedeni ise benzetim sırasında çok sayıda nedensellik hatası ile karşılaşılması ve bu hataların onarılmasının çok zaman almasıdır. Bu duruma örnek bir çalışma [36] kaynağında bulunabilir.

3. DENETİM NOKTASI MEKANİZMALARI (CHECKPOINTING)

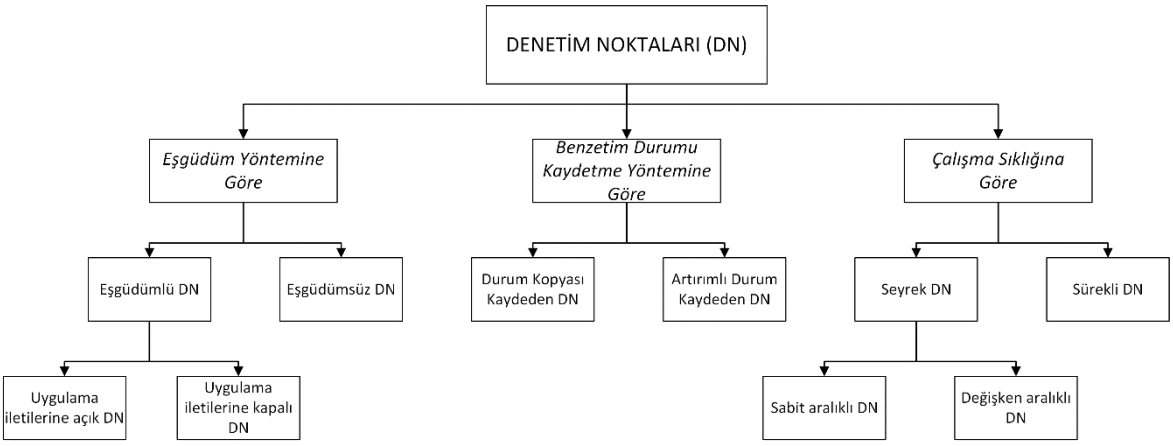
İyimser zaman yönetim mekanizmalarının başarımını etkileyen önemli konulardan bir tanesi de denetim noktalarıdır. Denetim noktaları, benzetime katılan Mİ'lerin zaman uyumlamasının yapıldığı ve benzetim durumunun kaydedildiği güvenli duraklardır. Tüm Mİ'ler belirlenen denetim noktasına ulaştığında benzetimin tutarlı bir durumda olduğu garantilenmiş olur. Bir başka deyişle artık hiçbir Mİ zaman damgası denetim noktasının zaman damgasından daha eski bir ileti almayacaktır. Bu nedenle, denetim noktasına ulaşıldığında benzetim durumu ileride ortaya çıkabilecek hatalara karşı kaydedilir. Denetim noktası mekanizmaları ile ilgili ayrıntılı bilgi [39] ve [40] çalışmalarında bulunabilir.

Denetim noktalarının kullanıldığı iyimser yaklaşımların çalışma mekanizması Şekil 3.1'de örneklendirilmiştir. Bu örnekte bir benzetim $Mİ_A$ ve $Mİ_B$ tarafından koşut olarak işletilmektedir. 13. zaman biriminde $Mİ_B$, $Mİ_A$'ya bir olay atamaktadır. Bu olayın $Mİ_A$ tarafından 15. zaman biriminde işletilmesi gerektiği; ancak ağdaki gecikmelerden dolayı bu iletinin $Mİ_A$'ya ancak 19. zaman biriminde ulaştığı varsayılsın. Bu iletinin zamanı $Mİ_A$'nın yerel zamanından daha eski olduğu için artık $Mİ_A$ bu iletiyi bir mızıkçı ileti olarak sınıflandıracak ve benzetimi onarma sürecine girecektir. Bu amaçla $Mİ_A$, daha önceden bilinen sağlıklı bir duruma geri dönmeli ve olayları doğru bir şekilde işleyerek durumunu onarmalıdır. Onarma işleminin birinci adımı benzetim durumunun denetim noktasında kaydedilen duruma çekilmesidir. İkinci adım, ileri sarma (İng. *coasting forward*) işlemi olarak adlandırılmakta olup denetim noktası ve mızıkçı ileti arasındaki olayların yeniden işletilmesidir. Bu adım sırasında diğer Mİ'lere herhangi bir ileti gönderilmez; çünkü bu iletiler zaten olağan işleyiş sırasında gönderilmiştir. İleri sarma işlemi tamamlandıktan sonra mızıkçı ileti olması gereken zamanda işleme alınır. Yeniden oynatma adı verilen dördüncü adımda ise benzetim mızıkçı iletiyi işledikten sonra yapması gereken işlemlere devam eder. Bu sırada Bölüm 2.2'de anlatıldığı üzere yine karşıt iletilerin oluşturulması ve ilgili Mİ'lere gönderilmesi gerekebilir.



Şekil 3.1. Denetim noktası kullanan Zaman Bükülmesi mekanizmasının işleyişi

Denetim noktası mekanizmaları farklı bakış açılarından ele alınarak çeşitli sınıflar altında incelenebilir. Şekil 3.2 bu tez kapsamında ele alınan denetim noktası yaklaşımlarının bir sınıflandırmasını sunmaktadır. Bu sınıflandırma [40] kaynağı temel alınarak oluşturulmuştur.



Şekil 3.2. Denetim Noktası yöntemlerinin sınıflandırılması

3.1. Eşgüdüm Yöntemine Göre Denetim Noktaları

Denetim noktası mekanizmaları, koşut çalışan Mİ'ler arasında eşgüdümlü (İng. *coordinated*) veya eşgüdümsüz (İng. *uncoordinated*) olarak çalıştırılabilir. Eşgüdümlü mekanizmalarda her bir Mİ aynı benzetim zamanında denetim noktasını işletir. Eşgüdümsüz mekanizmalarda ise Mİ'ler denetim noktasını aynı benzetim

zamanında işletmek zorunda değildir. Bu durum domino etkisi adı verilen olumsuzluğa yol açar. Domino etkisi, benzetim onarım işleminin bir önceki denetim noktasında yapılamaması ve geri sarma işleminin daha önceki benzetim zamanlarına yayılması anlamına gelmektedir [40].

Eşgüdümlü mekanizmalar, Mİ'lerin denetim noktasını işletirken uygulama iletilerine açık veya kapalı olmalarına göre de ikiye ayrılırlar. Uygulama iletilerine açık mekanizmalar mızıkçı iletilerin alınmasına da açık olacağı için denetim noktası işletilirken benzetimin geçmiş bir zamana dönmesine ve benzetim durumunun kaydedilmesinin yarıda kesilmesine neden olabilirler. Uygulama iletilerine açık mekanizmalar denetim noktası esnasında bir mızıkçı ileti alabilirler ve benzetimi onarmak durumunda kalabilirler. Ayrıca bu mekanizmalar güvenilir ve ilk-giren-ilk-çıkar sırasına uygun bir ağda çalışmak zorundadırlar.

3.2. Benzetim Durumu Kaydetme Yöntemine Göre Denetim Noktaları

Denetim noktaları, benzetim durumlarının güvenli bir şekilde kaydedildiği noktalardır. Bu kayıt işlemi o anki benzetim durumunun birebir kopyalanması ile gerçekleştirilebilir. Durum kopyalama yöntemi yüksek boyutlu verilerin kaydedilmesini gerektirebilir. Bu olumsuzlukları aşmak amacıyla artırımlı durum kaydetme yöntemleri önerilmiştir. Artırımlı yöntemler, kaydedilmesi gereken değişkenler üzerindeki güncellemeleri kaydeder ve onarıma ihtiyaç duyulduğunda tersine hesaplama (İng. *reverse computing*) yaparak istenilen geçmiş bir duruma dönülmesini sağlar. Ancak benzetim durumunun çok sık güncellendiği benzetimlerde durum kopyalama yöntemleri daha verimli olmaktadır [1]. Dolayısıyla artırımlı yöntem ve durum kopyalama yöntemlerinden herhangi birisinin diğerinden daha iyi olduğunu söylemek mümkün değildir. Bunun yerine benzetimi yapılacak modelin niteliğine göre hangi yöntemin seçileceğine karar vermek daha etkin sonuçlar getirecektir. Durum kaydetme yöntemlerine ait ayrıntılı bilgiler [1] ve [39] kaynağında bulunabilir.

3.3. Çalışma Sıklığına Göre Denetim Noktaları

Denetim noktalarının ne zaman işletileceğinin belirlenmesi iyimser zaman yönetim mekanizmalarının başarımını etkileyen en önemli ölçütlerden bir tanesidir. Bu

etkinin zaman ve bellek kullanımı açısından incelenmesi [41] çalışmasında yapılmıştır. Mİ'lerde gerçekleşen her olaydan sonra denetim noktası alınması en basit çözümlerden bir tanesidir. Ancak bu durumda denetim noktalarının çok sık işletilmesinden kaynaklı bir yavaşlama söz konusu olacaktır [42][43]. Bu nedenle seyrek denetim noktası (İng. *infrequent checkpointing, periodic checkpointing, sparse state saving*) alma yöntemleri önerilmiştir. Böylece denetim noktaları belli aralıklarla işletilecek ve sürekli durum kaydetme gibi bir olumsuzluk ortadan kaldırılmış olacaktır. Ancak bu durumda da mızıkçı ileti alan bir Mİ, istenilen geçmiş zamana doğrudan dönemeyecektir. Bunun yerine Şekil 3.1'de de gösterildiği gibi benzetim durumu daha da eski bir zamana çekilebilecektir. Ardından da denetim noktası zamanı ve mızıkçı ileti zamanı arasındaki olaylar yeniden işletilecektir (ileri sarma).

Seyrek denetim noktalarının olumsuz yönü denetim noktası aralıklarının belirlenmesi noktasında ortaya çıkmaktadır. Denetim noktası aralığı çok düşük olduğunda yine sık denetim noktalarındaki olumsuzluklar ortaya çıkar ve durum kaydetme işlemleriyle çok vakit kaybedilir. Aralık çok yüksek olduğunda ise bir mızıkçı ileti alındığında Mİ çok eski bir zamana dönmek zorunda kalabilir. Bu durumda da Şekil 3.1'de gösterilen ileri sarma ve yeniden oynatma adımları ile çok vakit kaybedilebilir. Bu nedenle en uygun denetim noktası aralığını belirlemek hem çok zor hem de modele son derece bağımlıdır. Üstelik en uygun denetim noktası aralığı benzetimin çalışması boyunca da değişiklik gösterecektir. Bu sorunu aşmak için denetim noktası aralığını dinamik olarak belirleyen ve kendini uyarlayabilen (self-adaptive) algoritmalar ortaya çıkmıştır. Dinamik denetim noktası aralıklarını kullanan yöntemlerin daha iyi sonuçlar ürettiği birçok çalışmada gözlendiği için bu yöntemler daha tercih edilebilir yöntemler olarak göze çarpmaktadır [42][43][44][45][46][47]. Bu tür yöntemler genellikle benzetimin son zamanlarındaki maliyet hesaplamalarını kullanarak bir sonraki denetim noktası aralığını belirlemeye çalışır. Bu maliyetler denetim noktası alma maliyeti, benzetim durumu kaydetme maliyeti, geri veya ileri sarma maliyeti olabilirler. Bu tür çalışmaların yanı sıra [48] kaynağında olduğu gibi denetim noktası sayısını sınırlı tutması gereken ve bu amaçla denetim noktası sıklığını belirlemeyi amaçlayan çalışmalara da rastlamak mümkündür. Gerçek zamanlı sistemler için en iyi denetim noktası sayısını belirlemeye yönelik bir çalışmaya da [49] kaynağında rastlanabilir.

4. İLGİLİ ÇALIŞMALAR

Koşut ve dağıtılmış benzetimlerde zaman yönetim mekanizmaları uzun bir süredir çalışılmaktadır. Son 30 yılda yapılmış literatür taramaları ve zaman yönetim mekanizması kavramlarının anlatıldığı çalışmalar [28], [50], [51] ve [52] kaynaklarında bulunabilir. İyimser zaman yönetim mekanizmasının kayda değer hızlanmalar sağladığı bazı çalışma örnekleri [2], [3], [38], [53] ve [54] kaynaklarında bulunabilir. [55] çalışmasında ise Zaman Bükülmesi mekanizmasının çeşitli gerçekleştirimlerinin bir karşılaştırması sunulmuştur.

Zaman Bükülmesi en çok bilinen iyimser zaman yönetim mekanizması olduğu için genellikle bu iki kavram günümüzde sıklıkla birbirinin yerine kullanılmaktadır. Ancak özellikle iyimser yaklaşımların yeni ortaya çıktığı zamanlarda farklı zaman yönetim mekanizmaları da önerilmiştir. Örneğin; “Breathing Time Buckets” isimli iyimser zaman yönetim mekanizması SPEEDES (Synchronous Parallel Environment for Emulation and Discrete-Event Simulation) işletim sistemi altında tanımlanmıştır [56]. [57] çalışmasında her iki algoritmanın karşılaştırılmasına yer verilmiştir. Bu karşılaştırmada “Breathing Time Buckets” algoritmasının daha iyi hızlanmalar sağlamasına karşın benzetim koşullarından daha fazla etkilendiği ve başarımının da bu koşullara daha bağımlı olduğu görülmüştür. Her iki yöntemin olumlu yönlerini bir araya getirmeye çalışan “Breathing Time Warp” algoritması da yine Steinman tarafından önerilmiştir [58]. [59] çalışmasında, zaman yönetim mekanizmasının davranışını benzetimin durumunu gözeterek dinamik bir şekilde değiştiren uyarlanabilir zaman uyumlama mekanizmalarının [60] Zaman Bükülmesi mekanizmasıyla karşılaştırılması yapılmıştır. Uyarlanabilir zaman uyumlama mekanizmalarının bir başka örneği de [61] kaynağında bulunabilir.

Bu bölümde, bu tez kapsamında yapılan çalışmaya benzer, literatürde yer alan çalışmaların bir derlemesi verilmiştir. Bu çalışmalar Zaman Bükülmesi mekanizmasını iyileştiren çalışmalar ve iyimser zaman yönetim mekanizmasını ETMB’de kullanan çalışmalar olarak aşağıdaki iki alt başlıkta incelenmiştir. Tez çalışması kapsamında yapılan çalışmaların literatürdeki çalışmalardan temel farkları da bir sonraki alt başlıkta ele alınmıştır.

4.1. Zaman Bükülmesi Mekanizmasının İyileştirilmesi Konusunda İlgili Çalışmalar

Zaman Bükülmesi mekanizması ilk olarak Jefferson tarafından önerilmiş [31] ve sonrasında birçok araştırmacı tarafından geliştirilen yeni yaklaşımlarla genişletilmiştir. Bu nedenle Zaman Bükülmesi'nin birçok farklı sürümüne rastlamak mümkündür. Örneğin fosil toplama ve "Virtual Time II" çalışmasında önerilen "cancelback" yaklaşımları, iletiler ve benzetim durumu için gerekli olan bellek alanlarının kullanımını iyileştirmeyi amaçlar [62]. Son yıllardaki göze çarpan bir örnek ise yine Jefferson tarafından önerilen ve sakıngan ve iyimser yaklaşımların olumlu yönlerini bir araya getirmeyi öneren "Virtual Time III" adlı çalışmadır [63]. Zaman Bükülmesi mekanizmasının başarımını artırmayı amaçlayan çalışmaların önemli bir kısmı geri sarma işlemlerinin sayısını veya getirdiği ek iş yükünü azaltmaya çalışmaktadır. Bunun nedeni geri sarma işleminin oldukça pahalı olabilmesidir. "Lazy cancellation" olarak adlandırılan yaklaşım geri sarma sayısını azaltan ve yaygın olarak bilinen yaklaşımlardan bir tanesidir [64][65][1]. Bu yaklaşım karşıt iletilerden kaynaklı olarak ortaya çıkan geri sarma işlemlerini önlemeye çalışır. [66] çalışmasında Zaman Bükülmesi'nin maliyetini azaltmak için dört farklı yöntem önerilmiştir. Bu yöntemlerden bir tanesi durum kaydetme işlemlerinin maliyetini, bir tanesi geri sarma işlemlerinin sayısını ve iki tanesi de geri sarma işlemlerinin maliyetini düşürmeyi amaçlar. Önerilen bu yöntemler sayesinde benzetimlerin daha hızlı sürelerde çalıştığı gösterilmiştir. "Lazy rollback" yaklaşımı ise bazı geri sarma işlemlerini ertelemeyi veya tamamen kaçınmayı önerir [33]. Ancak geçmiş durumları kaydetme maliyeti, anlık benzetim durumunun geçmiş durumla karşılaştırılması maliyeti (bir mızıkçı ileti alındığında geri sarma işleminin ertelenebilir veya kaçınılabilir olduğunu anlamak için bu karşılaştırmaya ihtiyaç duyulmaktadır) ve zaman sırası bozuk iletilerin işletilmesi bu yöntemin sorunlarından bazılarıdır. İlgili sorunlar Bölüm 6.1'de ayrıntılarıyla anlatılmıştır. Leong, Agrawal ve Agre benzetimdeki olayların sırasının değiştirilebilirliğini ve farklı sırada çalıştırılabilirliğini göz önünde bulundurarak geri sarma işlemlerinden kaçınmayı amaçlayan bir çalışma gerçekleştirmiştir [67]. Bir başka çalışmalarında ise iletilerin içeriğine göre hesaplamalarda kullanılması önemli olmayan mızıkçı iletilerin tespit edilmesi fikrini öne sürmüşlerdir [68]. Chen ve Szymanski "lookback" yaklaşımını önererek geri sarma işlemi sayısını azaltmayı hedeflemiştir [69][70]. Bu yaklaşım sayesinde bir

Mİ'nin başka Mİ'lerle iletişime geçmesine gerek kalmadan kendi yerel geçmişini değiştirmesi sağlanmıştır.

Geri sarma işlemlerinden kaçınmayı amaçlayan yöntemlerden bir tanesi de benzetim durumları için çoklu sürüm alma (İng. *multi-versioning*) tekniğinin kullanılmasıdır. Çoklu sürüm alma, Mİ'ler arasında paylaşılan değişkenlerin farklı benzetim zamanlarındaki değerlerini tutmayı amaçlar. Bu sayede bu değişkenlere ihtiyaç duyan Mİ'lerin değişkenin güncel veya önceki sürümlerine erişebilmesi mümkün olur [71]. Bu erişim sayesinde Mİ'ler geri sarma işlemlerinden kaçınabilirler. Çoklu sürüm alma yöntemi genellikle paylaşımlı bellek kullanan sistemlerde kullanılsa da [72], dağıtılmış bellek kullanan sistemlerde de kullanım örnekleri literatürde bulunabilir [73].

Zaman Bükülmesi mekanizmasının başarımını iyileştirmek için kullanılan yaklaşımlardan bir tanesi de olayların gerçekleşme zamanını daha gevşek bir mekanizma ile kontrol ederek geri sarma işlemlerini yapmamaktır. Bu konuya odaklanan çalışmalar genellikle benzetimin doğrulundan veya tekrarlanabilirliğinden belli oranlarda taviz verebilmektedir. Örneğin; [74] çalışmasında bazı mızıkçı iletiler göz ardı edilerek nedensellik kısıtları gevşetilmiş ve hatalı hesaplamalara izin verilmiş; ancak benzetimin dikkate değer oranda daha hızlı çalışması sağlanmıştır. Quaglia ve Baldoni'nin Mİ'lerin kendi içerisindeki koşutluğuna yönelik yaptığı bir çalışmada ise iyimser yaklaşımın olayları zaman sırasına uygun olmadan işletmesine izin verilmiştir [75]. Bu işletim, benzetimin zaman sıralı olarak işletilmesi durumunda ortaya çıkacak sonuçlara eşit olan benzetim koşumlarının kümesini gözeterek çalışmaktadır. Böylelikle eşdeğer olayların sıralaması farklı olsa da benzetimin sonucunun aynı olacağı durumlar ortaya konmakta ve geri sarma işlemlerinden kaçınılabilmektedir. Olay sırasını gevşeten çalışmalar arasında en bilinen yöntemlerden bazıları da zamansal belirsizlik (İng. *temporal uncertainty*) ve uzamsal belirsizlik (İng. *spatial uncertainty*) yöntemleridir. Fujimoto'nun önerdiği çalışmada sakıngan yaklaşıma sahip Mİ'lerin zamansal belirsizlik yönteminden yararlanmaları sağlanmıştır [76]. Böylece olayların kesin bir zamanda ortaya çıktığı değil, belirli bir zaman aralığında ortaya çıktığı varsayılmış ve olayların işletim sırası gevşetilmiştir. Sonuç olarak Mİ'lerin birtakım geri sarma işlemlerinden kaçınması sağlanmıştır. Bu durum benzetim sonuçlarında ihmal edilebilir seviyede hataya izin

vererek benzetimin daha hızlı çalışmasını sağlamıştır. [77] çalışmasında ise zamansal belirsizliğin benzetimlerin hızına ve doğruluğuna etkilerini inceleyen bir durum çalışması yapılmıştır. Bu çalışmada, benzetim sonuçları yaklaşık %15'lik bir hata ile hesaplanarak benzetim hızında önemli bir artış sağlanabildiği gösterilmiştir. Yakın zamanda ortaya çıkan bir başka durum çalışması da etmen tabanlı ve uzamsal koşut trafik benzetimlerinde belirsizliklerden yararlanmayı hedeflemiştir [22]. Mİ'ler arasındaki zaman uyumlamayı gevşeterek ileri bakma (İng. *lookahead*) değerini artırabilmek için benzetime bir miktar sezgisellik (İng. *heuristic*) sağlanmış; böylece trafik benzetimlerindeki belirsizliklerden yeterince yararlanılması sağlanmıştır. Bu çalışmadaki deneylerde benzetim hızında kayda değer iyileştirmeler sağlandığı ve istatistiksel benzetim sonuçlarının bozulmadığı gözlenmiştir. Zaman Bükülmesi mekanizması üzerinde zamansal belirsizliği kullanan bir çalışma da [78] kaynağında görülebilir. Zamansal belirsizliğin, etmen tabanlı ve Zaman Bükülmesi mekanizmasının kullanıldığı benzetimler için bir gerçekleştirimi de Beraldi ve çalışma arkadaşlarının yaptığı çalışmada bulunabilir [79]. Önceki çalışmalara benzer şekilde bu çalışmada da geri sarma işlemlerinin sayısının azaltılması sonucunda benzetimin hızlandırıldığı gösterilmiştir. Ayrıca modele özgü olarak tanımlanan belirsizlik zaman aralığının makul seviyelerde tutulmasıyla benzetim sonuçlarının doğruluğunun da riske girmediği belirtilmiştir. Zamansal belirsizliği Zaman Bükülmesi mekanizması üzerinde inceleyen bazı durum çalışmaları da [80] ve [81] kaynaklarında görülebilir. Bu çalışmalarda da benzetim hızının artırılabildiği ve benzetim sonuçlarının doğruluğunun istatistiksel olarak bozulmadığı gösterilmiştir. Quaglia ve Beraldi tarafından önerilen uzamsal belirsizlik tekniği de zamansal belirsizliğe benzer şekilde geri sarma işlemlerinin sayısının azaltılmasını amaçlar [82]. Uzamsal belirsizlik, benzetimdeki bir olayın kesin bir noktada gerçekleştiğini değil belirli bir bölgede gerçekleştiğini varsayarak ilerler. Dolayısıyla benzetimdeki olayların gerçekleştiği konum bilgilerini gevşeterek benzetimin çalışmasına izin verir. Bu sayede yine zamansal belirsizliğe benzer şekilde benzetim hızı artırılırken benzetim doğruluğundan bir miktar feda edilir.

Model geliştiriciler her zaman benzetim tekniklerinin ayrıntılarını bilmek zorunda değildir. Özellikle koşut ve dağıtılmış işlem gibi bilgisayar bilimlerinin alanına giren konularda model geliştiricilerin bilgi sahibi olmaması son derece doğaldır. Bu nedenle benzetimin nasıl koşturulacağının ayrıntılarını kullanıcıdan soyutlayabilen

çalışmalar, farklı disiplinlerden gelen araştırmacılar için önemli bir katkı sağlamaktadır. Bu alanda geniş bir literatür taraması ve birkaç özgün yöntem Pellegrini tarafından gerçekleştirilen tez çalışmasında bulunabilir [83]. O tezde önerilen çalışmalardan bir tanesi “Event and Cross-State synchronization” olarak adlandırılan ve programcıların benzetimdeki herhangi bir nesnenin durumuna erişebilmesini sağlayan olay işleyici (İng. *event-handler*) yazabilmesini sağlar. Önerilen bu yaklaşımın programlanabilirliği ve başarımı çok etmenli bir keşif modeli üzerinden [84] çalışmasında incelenmiştir. Pellegrini ve çalışma arkadaşlarının paylaşımlı bellek mimarileri için önerdiği bir diğer çalışma da [85] kaynağında bulunabilir. Bu çalışmada da otomatik olarak koşutlaştırılan bir benzetimde Mİ’lerin global değişkenlere erişebilmesi sağlanmaktadır. Böylece benzetimin kayda değer bir oranda hızlanma sağladığı görülmüştür. [86] ve [87] çalışmalarında benzetimlerin hızlı bir şekilde çalıştırılmasını ve bellek alanının daha verimli bir şekilde kullanılmasını sağlarken; benzetim durumu kaydetme işlemlerinin de kullanıcıdan soyut bir şekilde gerçekleştirilmesini sağlayan bazı çalışmalar görülebilir.

İyimser zaman yönetim mekanizmasını iyileştirmeyi amaçlayan çalışmaların önemli bir kısmı da paylaşımlı bellek mimarileri üzerine odaklanmıştır. [88] çalışmasında çoklu çekirdekli sistemler için “extended LPs” yaklaşımı önerilmiştir. Bu yaklaşımla, özel bir yetki verilen bazı Mİ’lerin, paylaşılan durum değişkenlerine erişebilmesine olanak verilmiştir. Böylece bu özel Mİ’ler, etkileşimi çok olan Mİ’ler arasındaki iletişimi azaltmayı ve buna bağlı olarak da geri sarma işlemlerinin sıklığını azaltmayı sağlar. Benzer bir çalışma [89] kaynağında verilen tez çalışmasında da mevcuttur. Bu çalışmada da Mİ’lerin diğer Mİ’lerin durum değişkenlerine erişmesine izin verilerek benzetim hızının artırılması sağlanmıştır. Yakın zamanda yayınlanan bir diğer çalışmada önerilen “share everything” konsepti ile de ciddi hızlanmalar sağlandığı görülmüştür [90]. [91] çalışmasında çoklu çekirdekli sistemlerde yüksek öncelikli olayların işlemcilere atanmasıyla nedensellik hatalarının daha az gerçekleşmesi amaçlanmıştır. Paylaşımlı bellek mimarisindeki çalışmalar genellikle yazılım tabanlı çözümleri önerse de donanım tabanlı çözümler öneren çalışmalara da rastlamak mümkündür. Örneğin; [92] çalışmasında çok çekirdekli bir sistemde işlemci çekirdeklerinin çalışma sıklıkları ayarlanarak Zaman Bükülmesi mekanizmasının hızlandırılabildiği gösterilmiştir.

Yukarıdaki yaklaşımların yanısıra iyimser zaman yönetiminin; birlikte çalışabilirlik yaklaşımları ile kullanımını sağlayan veya çeşitli yazılım ve donanım mimarileri için özelleştirilmiş gerçekleştirimini sağlayan çalışmalara da rastlamak mümkündür. Örneğin; HLA standardı kullanılan dağıtılmış benzetimlerde zaman yönetim mekanizmaları [93] kaynağında incelenmiştir. Bulut mimarilerinde Zaman Bükülmesi mekanizmasının kullanımına yönelik bir çalışma [94] kaynağında; zaman yönetim mekanizmalarının başarımına yönelik bir inceleme ise [95] kaynağında bulunabilir. [96] çalışmasında ise geçici dağıtılmış bir bilgisayar kümesi (İng. *Beowulf cluster*) üzerinde Zaman Bükülmesi mekanizmasının çalışması incelenmiştir. Son yıllarda yapılan bir çalışmada [97] Zaman Bükülmesi mekanizmasının GPU donanımlarındaki tasarımına ve gerçekleştirimine ait bir araştırma bulunabilir. [98] çalışmasında ise Zaman Bükülmesi'nin MPI ve çoklu-iş parçacıklı olarak gerçekleştirimine yönelik bir araştırma bulunabilir.

4.2. İyimser Zaman Yönetimi Yaklaşımının ETMB Çalışmalarında Kullanılması

Hem ETMB yaklaşımı hem de iyimser zaman yönetim mekanizmaları uzun zamandır modelleme ve benzetim dünyasında çalışılan konulardır. Ancak ETMB çalışmalarında zaman yönetim mekanizmalarının etkisini inceleyen çalışmaların sayısının sınırlı olduğu görülmüştür. Etmen tabanlı benzetimler için kullanılabilecek yazılımların ayrıntılı bir değerlendirmesi Bölüm 5.1'de verilmiştir.

Çok etmenli sistemlerin benzetimi için zaman yönetim desteği sağlanmasına yönelik bir çalışma [99] kaynağında bulunabilir. Bu çalışmada çok etmenli sistemler için geliştiriciden soyutlanmış bir zaman yönetim altyapısı önerilmiştir. [100] çalışmasında ise hem sakıngan hem de iyimser zaman yönetim mekanizmalarını destekleyen çok etmenli sistemlerin dağıtılmış benzetimi için bir uygulama çatısı örneği bulunabilir. [101] çalışmasında etmen tabanlı benzetimler için alana özgü (İng. *domain specific*) bilgileri de kullanarak Mİ'lerin aşırı iyimser davranmasını önleyen melez (İng. *hybrid*) bir zaman yönetim mekanizması önerilmiştir. Bir başka çalışmalarında da yazarlar, etmen tabanlı modellerin endüstriyel projelerde kullanılabilmesi için benzetimlerin mutlaka ölçeklenebilir olması gerektiğini; bunun da yalnızca iyimser zaman yönetim mekanizmalarıyla sağlanabileceğini savunarak ETMB çalışmaları için iyimser bir zaman uyumlama mekanizması önermişlerdir [102]. [103] çalışmasında MASON ortamında gerçekleştirilmiş seri bir modelin

otomatik bir dönüştürme yönteminden geçirilerek Zaman Bükülmesi mekanizmasını kullanan SaSSy yazılımında koşut bir şekilde koşturulması sağlanmıştır. Çok etmenli sistemlerin dağıtılmış benzetimi için iyimser zaman yönetimi mekanizması kullanan ve başarımını analiz eden çalışmalardan bazıları da [104], [105] ve [106] kaynaklarında bulunabilir.

Uzamsal koşut benzetimler için iyimser zaman yönetim mekanizmalarını kullanan çalışmalara da literatürde rastlamak mümkündür. Örneğin; [107] çalışmasında etmenler için etki çapı kavramı tanımlanmış ve çok etmenli dağıtılmış benzetimler için uyarlanabilir bir iyimser yaklaşım önerilmiştir. Bu çalışmada etmenlerin okuma-yazma örüntüleri ve geri sarma sıklıkları arasındaki ilişki ortaya konmuştur. Uzamsal ortamda koşturulan etmen tabanlı hastalık yayılım modelleri için Mİ'ler arasındaki ileti alışverişini azaltarak benzetim hızını artırmayı hedefleyen bir yöntem [19] çalışmasında önerilmiştir. Uzamsal etmen tabanlı modellerin sosyo-ekolojik sistemlerde kullanımına yönelik bir başka örnek de [108] kaynağında görülebilir. Uzamsal ve olasılıksal sistem benzetimlerinde geri sarma sayısını azaltmayı amaçlayan ve bu nedenle Mİ'lerin ne zaman kendilerini beklemeye alacağına karar vermeyi sağlayan bir mekanizma [24] çalışmasında önerilmiştir. [23] kaynağında da uzamsal ve olasılıksal benzetimlerdeki iyimserlik kontrolünü sağlayarak geri sarma sayısını azaltan bir çalışma sunulmuştur. Bu çalışmada işlemcilerin komşu işlemcilere atayacakları olayların zamanına ait kestirimlerini birbirleriyle paylaşması sağlanmıştır. Bu sayede uzamsal benzetimlerde kayda değer hızlanmalar elde edildiği görülmüştür.

4.3. Tez Çalışması Kapsamında Önerilen Yöntemlerin Literatürdeki Çalışmalardan Farkı

Önceki çalışmalarla karşılaştırıldığında bu tez çalışması şu yönleriyle bir fark ortaya koymaktadır:

- Önerilen alt-durum kaydetme ve öngörülmüş geri sarma mekanizmaları uzamsal-koşut etmen tabanlı benzetimler için genel amaçlı bir çözüm sunar ve modele özgü değildir.
- Önerilen alt-durum kaydetme ve öngörülmüş geri sarma mekanizmaları benzetimlerin çalışma süresini iyileştirmeyi amaçlarken benzetimlerin doğruluğu ve tekrarlanabilirliğinden taviz vermez. Her iki yöntem de

benzetimin sonucunun değişebileceği durumlarla karşılaştığında geri sarma işlemlerini yapmaktan kaçınmaz.

- Önerilen alt-durum kaydetme ve öngörülmüş geri sarma mekanizmaları zaman uyumlamayı gevşeterek doğruluktan taviz veren çalışmalarla birlikte de kullanılabilir. Zamansal ve uzamsal belirsizlik yöntemleri hem alt-durum kaydetme hem de öngörülmüş geri sarma mekanizmaları için kullanılarak benzetimlerin daha da hızlı çalıştırılması mümkündür.
- Önerilen alt-durum kaydetme ve öngörülmüş geri sarma mekanizmaları hem paylaşımlı bellek hem de dağıtılmış bellek mimarileri için çalışabilecek yöntemlerdir.
- Önerilen alt-durum kaydetme ve öngörülmüş geri sarma mekanizmaları büyük ölçüde model geliştiricilerden soyutlanmıştır. Her iki yöntem de etmenlerin yalnızca konumsal bilgilerini kullanarak çalışabilir. Ancak geliştirici modelin kullandığı etkileşim tanımlarını benzetim aracına tanımlarsa her iki yöntem de çok daha iyi bir başarım gösterebilir. Bu yöntemlerin kullanıcıdan tamamen soyutlanmış olduğunun söylenememesi de bu noktadan kaynaklanmaktadır.
- Denetim noktası sıklığını ayarlamak için kullanılan TAÇA yöntemi, literatürdeki algoritmaların kullandığı maliyet modellerine göre çok daha basit bir yöntem kullanır. Bu yöntem her zaman için en iyi sonucu vermese de mevcut dinamik yöntemlerden iyi sonuçlar verdiği durumlarla karşılaşılmaktadır.

5. REPAST HPC YAZILIMININ ZAMAN BÜKÜLMESİ ALGORİTMASI İLE GENİŞLETİLMESİ

5.1. Etmen Tabanlı Modelleme ve Benzetim Yazılımları

Etmen tabanlı yaklaşımla geliştirilen modellerin benzetimlerini yapan birçok yazılım bulunmaktadır. NetLogo [109], Mason [110] ve Repast Simphony [111] yaygın olarak kullanılan yazılımlardan bazılarıdır. Tez çalışmasının ilk aşamasında koşut ve dağıtılmış olarak YBH sistemlerinde çalışabilen, iyimser zaman yönetim mekanizmasını kullanan, etmen tabanlı bir modelleme ve benzetim yazılımına ihtiyaç duyulmuştur. Bu ihtiyaca yönelik yazılımların sayısının oldukça az olduğu görülmüştür. İlgili yazılımlar arasından açık kaynak kodlu ve bu çalışma kapsamında değerlendirilebilecek olan yazılımların özet bir karşılaştırması Çizelge 5.1'de verilmiştir. Bu yazılımlar değerlendirildiğinde tez kapsamındaki deneylerin Repast HPC [26] kullanılarak yapılmasına karar verilmiştir. Bunun nedeni Repast HPC'nin; modern YBH sistemlerinde ETMB çalışmalarına yönelik olarak geliştirilmesi, açık kaynak olarak sunulan kodların anlaşılabilir olması ve yeterince doküman ve kullanıcı desteğine sahip olmasıdır. Diğer yazılımlar ise şu nedenlerden dolayı tercih edilmemiştir:

- D-MASON yazılımı [112] ETMB çalışmalarının atıl bilgisayarlardan yararlanarak daha hızlı çalışabilmesi için MASON projesinin bir uzantısı olarak geliştirilmiştir. Bu tez çalışmalarına başlandığı tarihlerde D-MASON'ın henüz ilk sürümü yeni yayınlanmış ve yeterli olgunluk seviyesinde olmadığı değerlendirilmiştir.
- ROSS yazılımı [113] Georgia Tech Time Warp [114] yazılımının yeniden geliştirilmesi ile ortaya çıkmıştır; dolayısıyla yeterince olgun bir yazılım olduğu değerlendirilmiştir. Ancak, bu tezde hedef alınan etmen tabanlı benzetimlere yönelik bir arayüz sağlamamaktadır. Ayrıca C tabanlı olması nedeniyle büyük ölçekli modellerin ROSS'ta gerçekleştirilmesinin ve yönetilmesinin de oldukça güç olduğu değerlendirilmiştir.
- SaSSy [115][116] yazılımının dağıtılmış ve dağıtılmış olmayan sürümleri bulunmaktadır. Yazılımın en son 2006'da çıkmış sürümüne ulaşılabilmesi, dokümantasyonunun yetersiz olması ve son kullanıcılardan ziyade akademik seviyede kalmasından dolayı tercih edilmemiştir.

- WARPED yazılımı, bu tezde hedef alınan etmen tabanlı benzetimlere yönelik bir arayüz sağlamamaktadır. Ayrıca WARPED yazılımının yeterli doküman desteği olmadığı ve geçici dağıtılmış bilgisayar kümelerini (Beowulf cluster) hedeflediği görülmüştür [117].
- ROOT-Sim yazılımı [118] benzetimlerin çoklu-iş parçacıklı olarak koşut bilgisayar ortamlarında çalıştırılmasını sağlamakta olup YBH sistemlerine uygun olarak geliştirilmediği değerlendirilmiştir. Ayrıca ROOT-Sim, bu tezde hedef alınan etmen tabanlı benzetimlere yönelik bir arayüz sağlamamaktadır.
- MUSE yazılımının [119] bu tez çalışmalarına başlandığı tarihlerde henüz yeterli olgunluk seviyesinde olmadığı değerlendirilmiştir. Ayrıca MUSE yazılımının doküman desteğinin de yeterli seviyede olmadığı görülmüştür.

Bu yazılımların kullanılabilirliğinin istenilen seviyede olmaması nedeniyle, tez kapsamındaki deneyler için Repast HPC yazılımı seçilmiştir. Repast HPC, C++ dilinde yazılmış olup, koşut işlemlerin MPI kütüphanesiyle haberleşmesini sağlamaktadır. Repast HPC'nin seçilme nedenleri arasında şu özellikleri gösterilebilir;

- Açık kaynak kodlu olması,
- Esnek, genişletilebilir ve anlaşılabilir kod yapısı,
- Geliştirici ve dokümantasyon desteği,
- Çok büyük ölçekte koşut (İng. *massively parallel*) modellerin YBH sistemlerinde çalıştırılabilmesi için geliştirilmiş olması ve
- Doğrudan ETMB yaklaşımına yönelik model geliştirme olanağı sağlaması.

Bu tez kapsamında ilk iş olarak Repast HPC yazılımı Zaman Bükülmesi mekanizmasıyla genişletilmiştir. Yapılan bu genişletmeyle birlikte bu dokümanın ilerleyen bölümlerinde anlatılacak olan özgün yaklaşımlar için bir deney ortamı sağlanmıştır. Zaman Bükülmesi mekanizmasının Repast HPC'ye nasıl uygulandığını daha iyi aktarabilmek amacıyla öncelikle Repast HPC'nin çalışma ilkesi bu bölümde özetlenmiştir.

Çizelge 5.1. Tez kapsamında kullanılması değerlendirilen benzetim yazılımları

Yazılımın Adı	Geliştirici	Programlama Dili	İletişim Kütüphanesi	Lisans Durumu	İyimser Zaman Yönetim Desteği	ETMB Desteği
Repast HPC	Argonne National Library	C++	MPI	New BSD License	Yok	Var
D-MASON	ISIS Lab, The Department of Computer Science of University of Salerno	Java,	Java Message Service, mpiJava	Apache License, Version 2.0	Yok	Var
ROSS	Rensselaer Polytechnic Institute	C	MPI	BSD 3-Clause “New” or “Revised” License	Var	Yok
SaSSy	Distributed Simulation Laboratory, The University of Georgia	Java	Java RMI	U/D	Var	Var
MUSE	Parallel and Cloud Computing Laboratory, Miami University	C++	MPI	GNU General Public License	Var	Var
ROOT-Sim	High Performance and Dependable Computing Systems (HPDCS), Sapienza, University of Rome	C	MPI	GNU General Public License	Var	Yok
WARPED	High Performance Computing Laboratory, University of Cincinnati	C++	MPI	The MIT License	Var	Yok

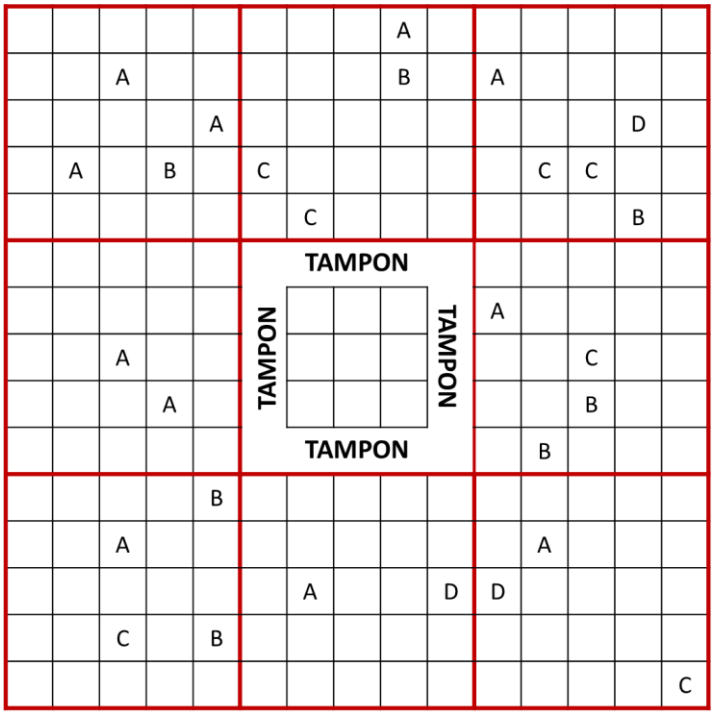
5.2. Repast HPC Çalışma İlkeleri

Repast HPC'de her bir Mİ ele alacağı olayların zamanlamasını benzetimin başında yapar. Böylece benzetim boyunca tekrarlanacak olaylar (Örn; etmenlerin davranışları), tek seferlik olaylar (Örn; benzetimin özel bir zamanında çalışması istenen olaylar) ve benzetim sonlanınca çalıştırılması istenen olaylar (Örn; sonuçların kaydedilmesi) benzetim başlamadan önce zamanlanır. Ayrıca benzetim sırasında da yeni olayların zamanlanması mümkündür. Zamanlanan olaylar her bir Mİ'nin olay kuyruğunda tutulur ve sırası geldiğinde Mİ tarafından çalıştırılır.

Repast HPC'nin mevcut gerçekleştiriminde sıkı bir sakıngan zaman yönetimi uygulanmaktadır. Tüm Mİ'ler yerel benzetim zamanını zaman uyumlu bir şekilde ilerletirler. Zaman uyumlama Mİ'lerin MPI kütüphanesinin "all-reduce" işlevini çağırmasıyla gerçekleştirilir. Bu işlevin çağırılmasından sonra yerel benzetim zamanı en küçük olan Mİ'ler sıradaki olayı işletirler. Bu nedenle en küçük yerel benzetim zamanı, aynı zamanda genel benzetim zamanı olarak da düşünülebilir. Mİ'ler arasındaki bu zaman uyumlama benzetimin yavaşlamasına da neden olan önemli etkenlerden bir tanesidir. Repast HPC geliştiricileri, karmaşık hesaplamalardan kaçınmak amacıyla ilk sürümlerde sakıngan yaklaşımı tercih ettiklerini; ancak daha gelişmiş zaman yönetim mekanizmaları için de açık kapı bıraktıklarını belirtmişlerdir [26].

Repast HPC etmenlerin birbirleriyle etkileşimi için iki tür ortam sunar: uzamsal ortam ve ağ ortamı. Uzamsal ortam 2 boyutlu bir ızgara veya bir sürekli uzay olabilmektedir. Repast HPC, benzetimin koşut çalıştırılmasını bu ortamları Mİ'lere eşit bir şekilde paylaştırarak gerçekleştirebilmektedir. Bir başka deyişle Repast HPC uzamsal-koşut olarak çalışabilmektedir. Böylece her bir Mİ, tüm benzetim uzayındaki belirli bir alt-uzayın sorumluluğunu üstlenir ve bu kısım üzerindeki etmenlerin eylemlerini gerçekleştirmesini sağlar. Etmenler uzay üzerinde belli koordinatlara sahip olup komşuluk üzerinden birbirleriyle etkileşime girerler. Bir Mİ; kendi sorumluluğunda olan uzayın sınırlarına yakın etmenlerin, komşu Mİ'lerdeki etmenlerle etkileşimi için diğer Mİ'lerle haberleşmek zorundadır. Bu nedenle Mİ'lerin sınırları komşu Mİ'lere de görünür olmalıdır. Bu görünür alanlar Repast HPC geliştiricilerince tampon (İng. *buffer*) alan olarak adlandırılmaktadır. Tampon alanlar için Mİ'lerin zaman uyumlamasının sağlanması geliştiricilerin sorumluluğundadır. Bir

başka deyişle geliştirici, Mİ'deki eylemlerin tamamlanmasının ardından zaman uyumlama işlevlerinin çağrılmasını sağlamalıdır. Uzamsal ortamların paylaşılmasını ve tampon alanlarını gösteren örnek bir görsel Şekil 5.1'de verilmiştir.



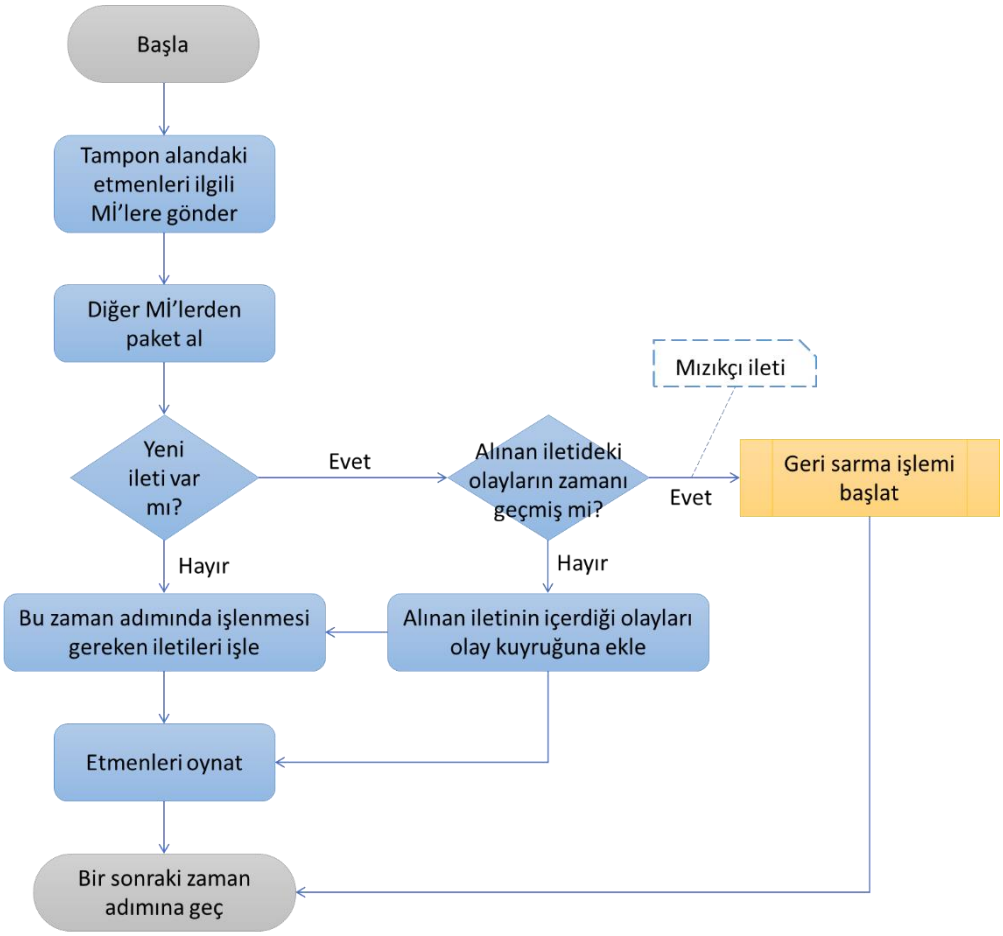
Şekil 5.1. Repast HPC'de 9 mantıksal işlem tarafından yönetilen iki boyutlu bir ızgara ortamı

Repast HPC'deki ağ ortamı ise bir çizge yapısı gibi düşünülebilir. Bu yapıda etmenler köşelere, etmenler arası bağlantılar ise kenarlara karşılık gelmektedir. Etmenler arası etkileşim kenarlar aracılığıyla gerçekleştirilir. Ağ ortamı, uzamsal ortamın üzerine inşa edilerek, doğrudan iletişim kurması gereken etmenler arasındaki etkileşimi kolaylaştırmak için kullanılabilir. Bu tez çalışmasında yalnızca uzamsal ortam üzerindeki etkileşimler ele alınmış olup, ağ ortamı kapsam dışı bırakılmıştır.

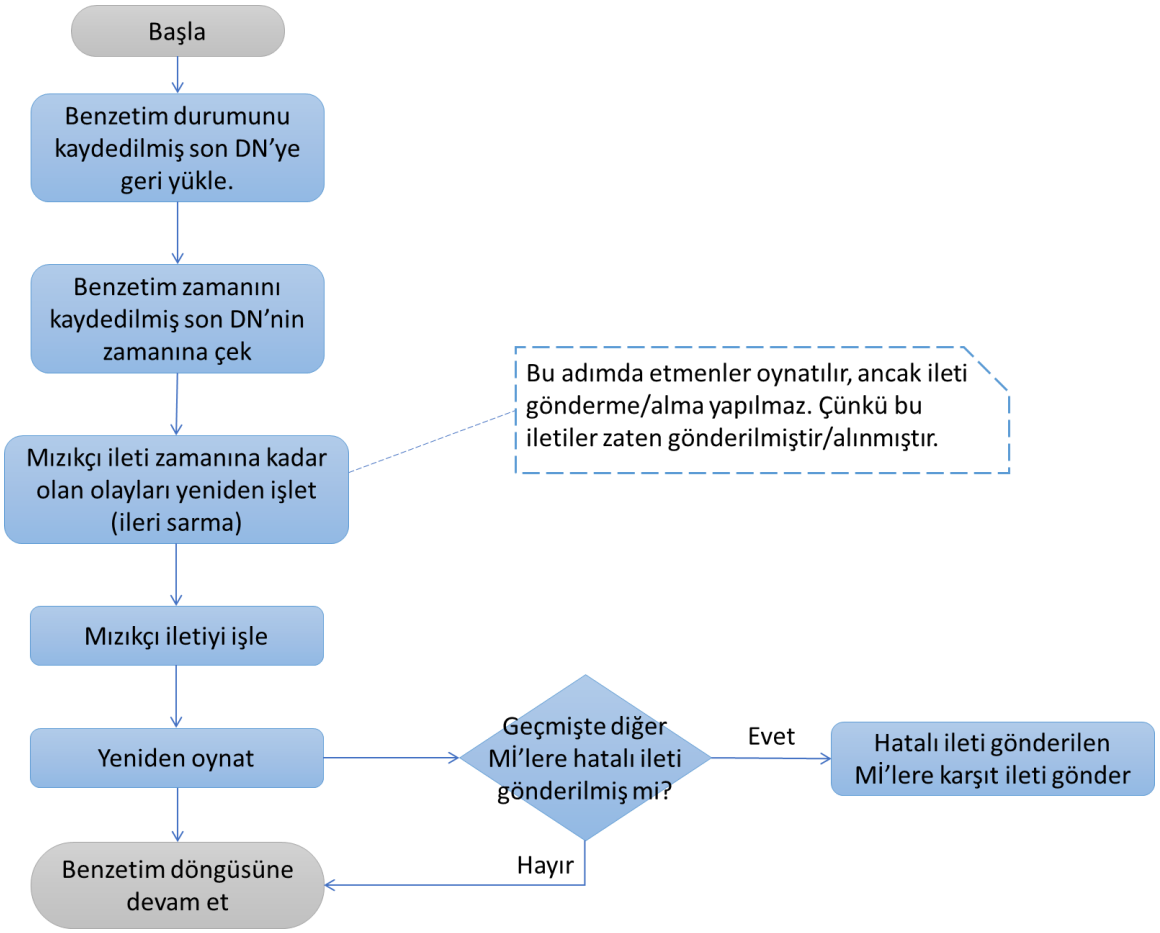
5.3. Benzetim Ortamı Mimarisi

Bir önceki bölümde açıklandığı üzere Repast HPC, olayların işletilmesi ve etmenlerin paylaşılması sırasında Mİ'lerin zaman uyumlama yapmasını sağlar. Olayların işletilmesi sırasındaki zaman uyumlama bir Mİ'nin diğerlerinin ilerisinde gitmesini önler. Böylece Mİ'ler yerel zamanlarını birlikte ilerletirler. Bir diğer zaman

uyumlama ise etmenlerin doğru zamanda komşu Mİ'lerle paylaşılması için yapılır. Her iki zaman uyumlama da benzetimin tutarlı olmasını ve olayların doğru sırada çalıştırılmasını sağlasa da koşutluğun azalmasına ve benzetimin yavaşlamasına neden olmaktadır. Zaman Bükülmesi mekanizmasının Repast HPC'de gerçekleştirilmesi ile zaman uyumlamaların azaltılarak benzetimlerin hızlandırılması amaçlanmıştır. Zaman Bükülmesi ile çalışan Repast HPC'de (RHPC_TW) Mİ'lerin akış şeması Şekil 5.2'de ve bir geri sarma işleminin ele alınışı Şekil 5.3'te verilmiştir. Bu tez çalışması kapsamında geliştirilen RHPC_TW'nin sınıf şemasının özet bir hali Ek - 3'te verilmiştir.



Şekil 5.2. RHPC_TW'de bir mantıksal işlemin akışı



Şekil 5.3. RHPC_TW'de bir geri sarma işleminin ele alınışı

5.3.1. Dolaylı Olaylar

Mİ'ler arasındaki etkileşimi daha iyi anlatabilmek amacıyla Repast HPC'deki olaylar iki sınıfta incelenebilir. Bunlardan ilki yukarıda anlatılmış olan ve modelleyici tarafından ne zaman çalışacağı belirlenen olaylardır. Öte yandan tampon alanlardaki etmenlerin doğru zamanda ilgili Mİ'lere bildirimi için Mİ'ler birbirlerine olay atayabilirler. Tez çalışmasının kalan kısmında bu olaylar dolaylı olay olarak adlandırılacaktır. Dolaylı olaylar RHPC_TW'nin en önemli noktalarını oluşturmakta olup şu üç başlıkta ele alınmıştır:

- Etmenlerin sahipliği: Bir etmen hareketi sırasında komşu bir Mİ'nin yönettiği alt-uzaya girebilir. Bu durumda ilgili etmenin sahibi olan Mİ güncellenmelidir.
- Etmenlerin komşu Mİ'lere görünürlüğü: Tampon alandaki etmenlerin komşu Mİ'ler tarafından da görülmesi sağlanıp, komşu Mİ'lerin yaptığı hesaplamalara dahil edilmesi sağlanmalıdır.

- Etmenleri güncelleme yetkisi: Tampon alanda yer alan etmenlerin, komşu bir Mİ'de yapılan hesaplamalar sonrası güncellenmesi gerekebilir. Bu durumda komşu Mİ'ler, güncellenecek etmenlerin sahibi olan Mİ'leri güncelleme isteği konusunda bilgilendirmelidirler.

Repast HPC dolaylı olayları zaman uyumlu olarak ele alır. RHPC_TW'de bu olayların zaman uyumlama yapılmadan ele alınması sağlanmıştır. Böylece Mİ'ler bir sonraki denetim noktasına kadar zaman uyumlama olmadan yerel benzetim zamanlarını ilerletebilirler. Ancak, bir Mİ mızıkçı bir ileti aldığında bir geri sarma işlemi başlatarak benzetim durumunu onarmak zorunda kalır.

5.3.2. RHPC_TW'de Kullanılan Denetim Noktası Mekanizması

RHPC_TW'de uygulama iletilerine açık, durum kopyası kaydeden ve değişken aralıklı olarak çalıştırılan bir denetim noktası yöntemi tercih edilmiştir. Bu tercihin nedenleri aşağıda verilmiştir:

- RHPC_TW'de yapılan deneyler güvenilir bir ağda çalıştırıldığı için uygulama iletilerine açık bir denetim noktası mekanizması seçilmiştir. Böylece bir denetim noktasının başlatılması için uygulama iletilerinin tamamlanması beklenmemektedir.
- Etmen tabanlı benzetimlerde, benzetim durumu çok sık güncellendiği için durum kopyası kaydeden bir denetim noktası mekanizması tercih edilmiştir. Artırımlı durum kaydeden denetim noktası yaklaşımlarının, benzetim durumunun çok sık değişimi karşısında daha başarılı olmayacağı değerlendirilmiştir.
- Literatürdeki birçok çalışmada da görüldüğü üzere değişken aralıklı denetim noktası mekanizmaları, hem sabit aralıklı denetim noktası mekanizmalarına hem de sürekli denetim noktası mekanizmalarına göre daha başarılı olduğu için değişken aralıklı bir denetim noktası mekanizması tercih edilmiştir [42][43][44][45][46][47].

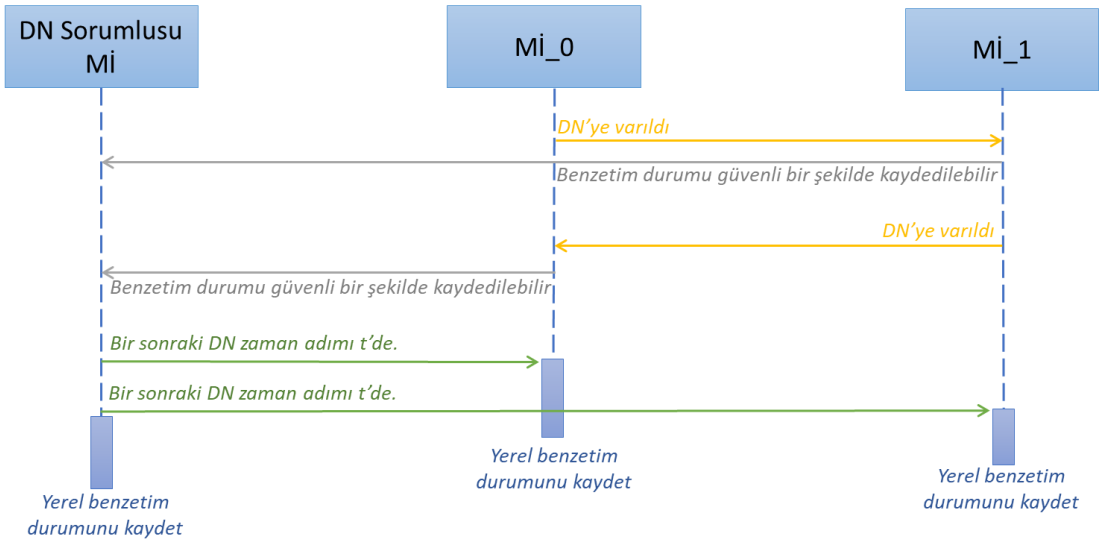
RHPC_TW'de denetim noktalarının sıklığı dinamik ve kendini uyarlayabilir (İng. *self-adaptive*) bir algoritma tarafından benzetim sırasında ayarlanmaktadır. Toplamsal Artış Çarpımsal Azalış (TAÇA) adı verilen ve testere dişlisi modeli olarak da bilinen

algoritma modelleme ve benzetim alanı dışında halihazırda kullanılmakta olan bir algoritmadır. Örneğin; bilgisayar ağlarında tıkanıklık kontrolü sürecinde bu algoritma kullanılmaktadır [27]. Anlaması kolay ve ölçülebilir yapısından dolayı RHPC_TW'de de bir sonraki denetim noktasının ne zaman alınacağı TAÇA algoritmasına göre belirlenmektedir. Benzetim başlangıcında denetim noktası sıklığı 1 olarak seçilir ve rastgele bir Mİ, denetim noktası sorumlusu olarak belirlenir. Her bir denetim noktası tamamlandığında denetim noktası sorumlusu TAÇA algoritmasını çalıştırır. Buna göre, bir önceki denetim noktası ile şu an alınan denetim noktası arasında herhangi bir geri sarma işlemi gerçekleşmediyse denetim noktası sıklığı artış sabiti kadar artırılır; gerçekleştiyse denetim noktası sıklığı azalış çarpanı ile çarpılır. Bu sayede bir nedensellik hatası ortaya çıkması durumunda, Mİ'ler arasındaki zaman uyumlamanın daha sıkı yapılması ve Mİ'lerin daha sakıngan davranması sağlanmış olur. TAÇA yöntemi matematiksel olarak (1) eşitliğindeki gibi ifade edilebilir. Bu eşitlikte χ_t, t anındaki denetim noktası aralığını, α artış sabitini, β ise azalış çarpanını göstermektedir. Bu tez kapsamında yapılan tüm deneylerde artış sabiti 1, azalış çarpanı ise 0.5 olarak belirlenmiştir.

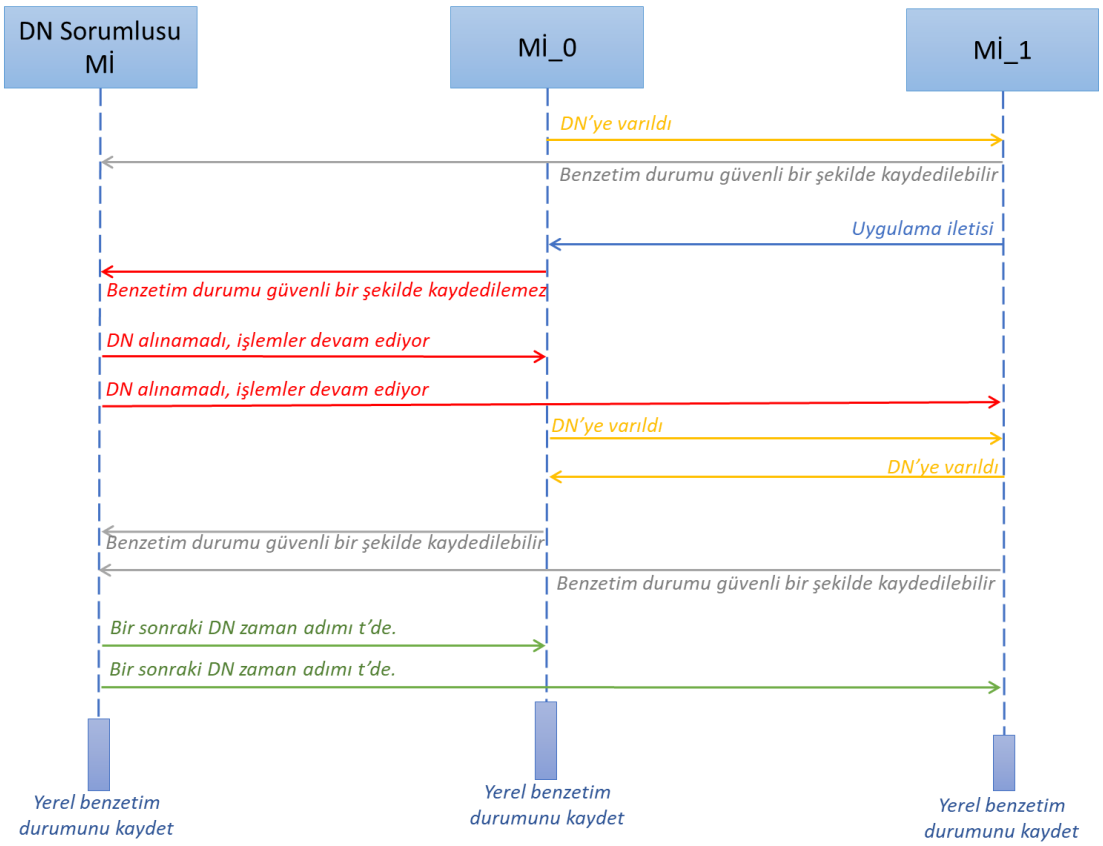
$$\chi_{t+1} = \begin{cases} \chi_t + \alpha, & bir\ önceki\ DN'dan\ beri\ bir\ nedensellik\ hatası\ gözlenmediyse \\ \chi_t \cdot \beta, & bir\ önceki\ DN'dan\ beri\ bir\ nedensellik\ hatası\ gözlendiyse \end{cases} \quad (1)$$

RHPC_TW'de kullanılan iki aşamalı bir denetim noktası mekanizmasının çalışması Şekil 5.4 ve Şekil 5.5'te gösterilmiştir. Bu gerçekleştirime göre benzetime katılan Mİ'lerden bir tanesi benzetimin başında denetim noktası sorumlusu olarak seçilir. Başlangıçta tüm Mİ'ler denetim noktası alma sorumluluğunun hangi Mİ'de olduğunu ve ilk denetim noktasının zamanını bilirler. Bir denetim noktası üç aşamada alınır. İlk aşamada denetim noktası zamanına ulaşan her Mİ, komşusu olan Mİ'lere denetim noktasına ulaştığını bildirir. Bu bildirim aynı zamanda bildirimi yapan Mİ'nin, denetim noktası bitene kadar komşularına bir uygulama iletisi göndermeyeceğinin güvencesini verir. İkinci aşamada tüm komşularının denetim noktasına ulaştığını öğrenen Mİ'ler denetim noktası sorumlusuna benzetim durumu kaydetmenin güvenli olduğunu söyler. Bir Mİ, tüm komşularından bu iletiyi alamazsa, denetim noktası sorumlusuna benzetim durumunu kaydetmenin güvenli olmadığını söyler. Üçüncü aşamada denetim noktası sorumlusu, tüm Mİ'lerden benzetim durumunu kaydetmenin güvenli olduğu bilgisini alırsa yeni denetim noktası sıklığının ne zaman

alınacağı bilgisini hesaplar. Sonrasında da bu bilgi tüm Mİ'lere gönderilir ve tüm Mİ'ler yerel benzetim durumunu kaydederek benzetimi işletmeye devam eder. Eğer denetim noktası sorumlusu tüm Mİ'lerden benzetim durumunun kaydedilmesinin güvenli olduğu bilgisini alamazsa önceki iki adım tekrar edilir.



Şekil 5.4. Denetim noktasının başarılı bir şekilde alınması



Şekil 5.5. Denetim noktası sırasında bir uygulama ileti alınması

TAÇA yöntemi benzetimin son zamanlarında geri sarma işlemi gerçekleşip gerçekleşmediğini göz önünde bulundurur. Bunun nedeni ise uzamsal benzetimlerde geri sarma işleminin Mİ'lerin tampon alanında bulunan etmenlerden kaynaklı olmasıdır. Bu etmenler ileride gerçekleşebilecek geri sarma işlemlerine dair bir ipucu verdiği için TAÇA yöntemi tercih edilmiştir.

5.4. Deney Ortamı

RHPC_TW'nin başarımını Repast HPC ile karşılaştırabilmek için bazı deneyler gerçekleştirilmiştir. Bu deneyler için ODTÜ-TSK MODSİMMER'de geçici bir dağıtılmış bilgisayar kümesi (Beowulf cluster) kurulup, benzetim ortamı bu dağıtılmış ağ üzerinde çalıştırılabilir hâle getirilmiştir. Bu ağ, özdeş olmayan 6 bilgisayar ve toplamda 22 işlemci çekirdeğine sahiptir. Her bir bilgisayar üzerinde Ubuntu 12.04 işletim sistemi çalıştırılmıştır. Bilgisayarlar bir yerel ağ üzerinden birbirlerine bağlanmıştır. Ağdaki bilgisayarlara ait özet bilgi Çizelge 5.2'de verilmiştir.

Çizelge 5.2. ODTÜ-TSK MODSİMMER'de kurulan dağıtılmış benzetim ortamı

İşlemci Modeli	İşlemci Frekansı	Çekirdek Sayısı	Bellek
Intel Xeon W3680	3,33 – 3,60 GHz	6	4 GB
Intel Xeon W3680	3,33 – 3,60 GHz	6	4 GB
Intel Core 2 Duo E7400	2.80 GHz	2	4 GB
Intel Core 2 Duo E7400	2.80 GHz	2	4 GB
Intel Core 2 Duo E7400	2.80 GHz	2	2 GB
Intel Core 2 Quad Q8400	2.66 GHz	4	4 GB

5.5. Durum Çalışması

"The Civil Violence" isimli model [120] Repast HPC ve RHPC_TW'de gerçekleştirilerek bir durum çalışması yapılmıştır. Böylece sakıngan ve iyimser zaman yönetim mekanizmalarının başarımı bir etmen tabanlı benzetim üzerinde gözlenmiştir. Hem gerçekleştirme kolaylığı hem de farklı parametreler kullanarak zaman yönetim mekanizmasının farklı koşullar altındaki başarımını gözleme olanağı sağlamasından dolayı tez çalışmasında bu model kullanılmıştır. Bu modeli kısaca özetlemek gerekirse;

- Modelin amacı insanların çeşitli parametrelere (örn; etraftaki sakin, eylemci ve polislerin sayısı, hükümet baskısı, eyleme katılma eşiği, yakalanma olasılığı vb.) göre toplumsal olaylara nasıl tepki verdiğinin, hangi şartlarda eylemlere katıldığının ve bu eylemlerin nasıl kontrol altına alındığının modellenmesidir.
- Dört çeşit etmen bulunmaktadır: eylemci (activist), sakin (quiet), polis (cop) ve mahkum (jailed).
- Etmenler 2 boyutlu bir ızgara üzerinde etrafındaki 8 hücreye hareket edebilmektedir.
- Eylemciler polislerden kaçmaya çalışmaktadır. Bir eylemci polis tarafından yakalandığında mahkuma dönüşmektedir.
- Mahkumlar cezaları bitinceye kadar bulundukları hücreden hareket edememekte ve diğer etmenlerle etkileşime girememektedir. Cezası biten bir mahkum bir sakin etmene dönüşmektedir.
- Sakinler ızgara üzerinde dolanmakta ve eyleme katılıp katılmama kararını vermektedirler. Bu karar mekanizması [120] kaynağında verilen matematiksel modele dayanmaktadır.
- Polisler etraflarındaki eylemcileri yakalamaya çalışmaktadır

Deneyler 120×120'lik ve 360×360'lık bir ızgara ortam üzerinde ve farklı işlem yükleriyle 500 benzetim adımı boyunca koşturulmuştur. Benzetimlerin başlangıcında etmenler ızgaraya rastgele olarak ve ızgaranın %75'ini dolduracak şekilde yerleştirilmiştir. Yani, 120×120'lik ızgarada toplam 10800, 360×360'lık ızgarada ise toplam 97200 etmenin benzetimi yapılmıştır. Etmenlerin komşuluk aralığı 1 hücre seçilmiştir. Yani iki etmenin birbiriyle etkileşime girebilmesi için bulundukları hücrelerin komşu olması gerekmektedir.

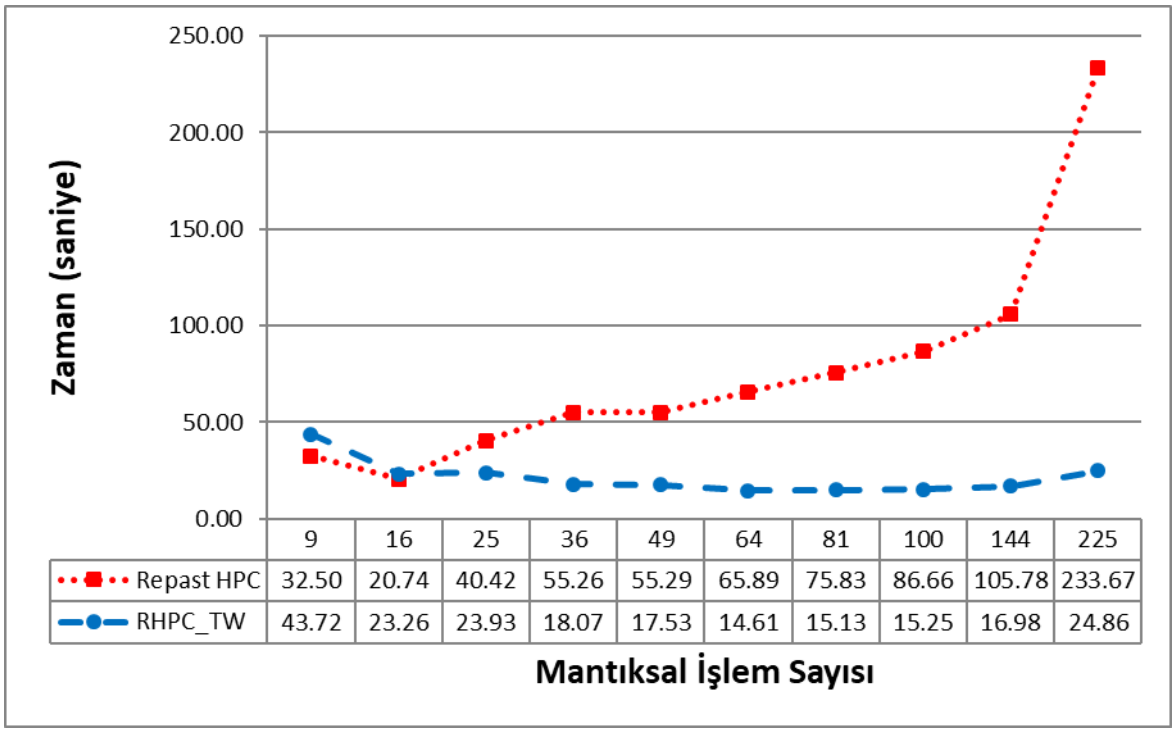
Mİ sayısı 9, 16, 25, 36, 49, 64, 81, 100, 144 ve 225 olarak seçilmiş olup; çekirdek sayılarına göre işlemcilere dağıtılmıştır. Mİ sayılarının bu şekilde seçilmesinin nedeni, etmen sayılarını Mİ'lere sistematik bir şekilde atayabilmektir.

5.6. Repast HPC ve RHPC_TW'nin Karşılaştırılması

Repast HPC ve RHPC_TW'de aynı modeller koşturulmuş ve başarımları hızlanma açısından karşılaştırılmıştır. Deneyler farklı konfigürasyonlar için en az 4 kez tekrarlanmış ve elde edilen ortalama değerler sunulmuştur. Çizelge 5.3, Şekil 5.6 ve Şekil 5.7'de 120×120'lik ızgara ortamında yapılan benzetimlerin tamamlanması için geçen süre, bu sürelerin standart sapmaları ve hızlanma değerleri verilmiştir. Çizelge 5.4, Şekil 5.8 ve Şekil 5.9'da ise 360×360'lık ızgara ortamında yapılan benzetimlerin tamamlanması için geçen süre, bu sürelerin standart sapmaları ve hızlanma değerleri verilmiştir. Hızlanma değeri, $S = \frac{T_S}{T_P}$ eşitliğine göre hesaplanmıştır [121]. Bu eşitlikte S hızlanma değerini; T_S benzetimin tek bir Mİ'de çalıştırıldığında geçen zamanı; T_P ise benzetimin koşut çalıştırıldığında geçen zamanı göstermektedir. Şekil 5.7 ve Şekil 5.9'da verilen ideal hızlanma değerleri fiziksel çekirdek sayısı üzerinden verilmiştir.

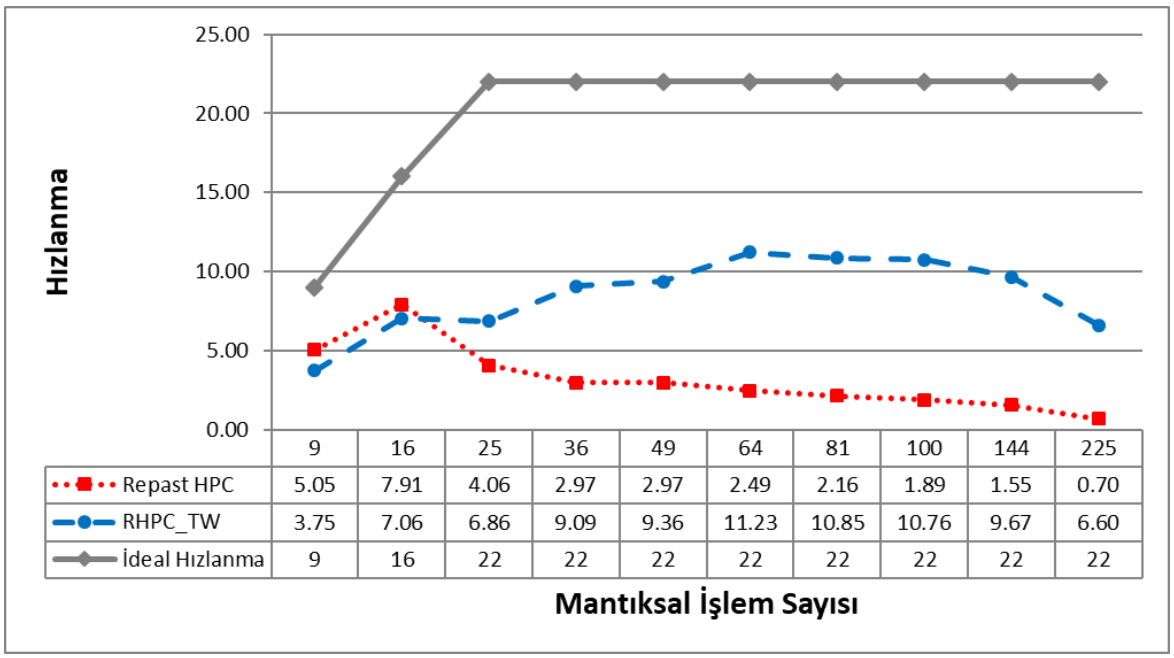
Çizelge 5.3. 120×120'lik ızgara ortamında farklı sayıdaki mantıksal işlemlerde benzetimin çalıştırılma süresi

		Mİ sayısı									
		9	**16**	**25**	**36**	**49**	**64**	**81**	**100**	**144**	**225**
Repast HPC	**Geçen zaman [s]**	32,5	20,7	40,4	55,3	55,3	65,9	75,8	86,7	105,8	233,7
	Standart sapma (σ) [s]	0,1	0,3	0,3	0,5	0,7	0,9	0,4	0,4	1,1	4,2
RHPC_TW	**Geçen zaman [s]**	43,7	23,3	24,0	18,1	17,5	14,6	15,1	15,3	17,0	24,9
	Standart sapma (σ) [s]	0,4	0,3	0,33	0,3	0,2	0,2	0,2	0,1	0,2	0,9



	9	16	25	36	49	64	81	100	144	225
Repast HPC	32.50	20.74	40.42	55.26	55.29	65.89	75.83	86.66	105.78	233.67
RHPC_TW	43.72	23.26	23.93	18.07	17.53	14.61	15.13	15.25	16.98	24.86

Şekil 5.6. 120×120'lik ızgara ortamında farklı sayıdaki mantıksal işlemlerde benzetimin çalıştırılma süresi

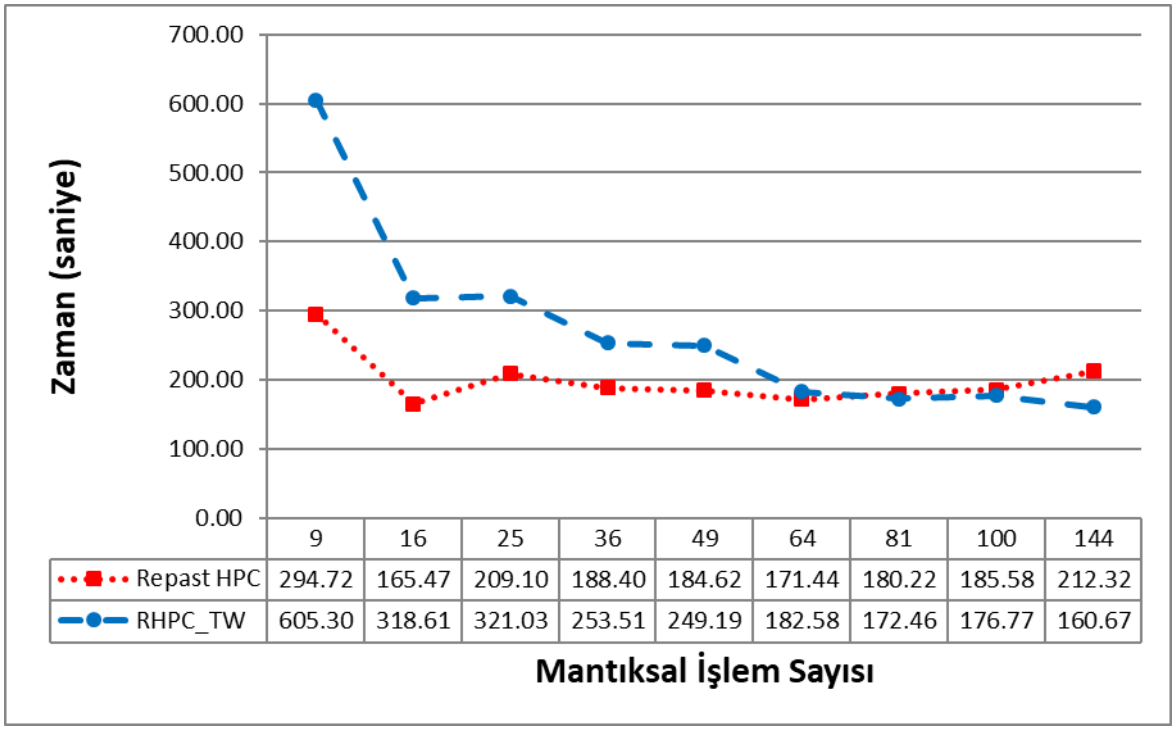


	9	16	25	36	49	64	81	100	144	225
Repast HPC	5.05	7.91	4.06	2.97	2.97	2.49	2.16	1.89	1.55	0.70
RHPC_TW	3.75	7.06	6.86	9.09	9.36	11.23	10.85	10.76	9.67	6.60
İdeal Hızlanma	9	16	22	22	22	22	22	22	22	22

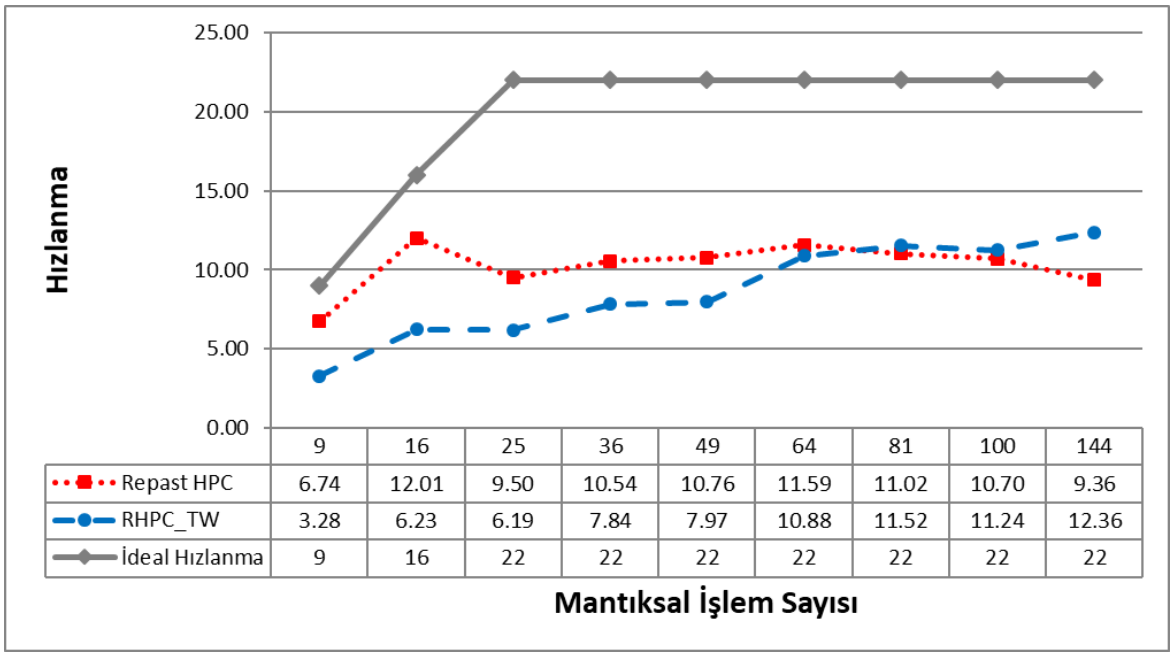
Şekil 5.7. 120×120'lik ızgara ortamında farklı sayıdaki mantıksal işlemlerde elde edilen hızlanma değeri

Çizelge 5.4. 360×360'lık ızgara ortamında farklı sayıdaki mantıksal işlemlerde benzetimin çalıştırılma süresi

		Mİ sayısı								
		9	**16**	**25**	**36**	**49**	**64**	**81**	**100**	**144**
Repast HPC	**Geçen zaman [s]**	32,5	20,7	40,4	55,3	55,3	65,9	75,8	86,7	105,8
	Standart sapma (σ) [s]	0,1	0,3	0,3	0,5	0,7	0,9	0,4	0,4	1,1
RHPC_TW	**Geçen zaman [s]**	43,7	23,3	23,9	18,1	17,5	14,6	15,1	15,3	17,0
	Standart sapma (σ) [s]	0,4	0,3	0,3	0,3	0,2	0,2	0,2	0,1	0,22



Şekil 5.8. 360×360'lık ızgara ortamında farklı sayıdaki mantıksal işlemlerde benzetimin çalıştırılma süresi



	9	16	25	36	49	64	81	100	144
Repast HPC	6.74	12.01	9.50	10.54	10.76	11.59	11.02	10.70	9.36
RHPC_TW	3.28	6.23	6.19	7.84	7.97	10.88	11.52	11.24	12.36
İdeal Hızlanma	9	16	22	22	22	22	22	22	22

Şekil 5.9. 360×360'lık ızgara ortamında farklı sayıdaki mantıksal işlemlerde elde edilen hızlanma değeri

120×120'lik ızgarada Mİ sayısı düşükken sakıngan zaman yönetim mekanizması, iyimser zaman yönetim mekanizmasından daha hızlı bir şekilde benzetimi tamamlamıştır. Ancak Mİ sayısı arttıkça iyimser zaman yönetim mekanizmasının, sakıngan zaman yönetim mekanizmasını yakaladığı ve daha iyi sonuçlar verdiği görülmüştür. Her iki zaman yönetim mekanizmasında da Mİ'lerin belli bir sayıyı geçmesinden sonra hızlanma değerlerinin Amdahl yasasının olumsuz etkisiyle (koşut çalışan işlemci sayısını artırdıkça iletişim işlemlerinin baskın gelmesi ve başarımın düşmesi) [122] karşılaştığı gözlenmiştir. Ancak bu etki iyimser zaman yönetim mekanizmasında daha geç görülmeye başlanmıştır. Sakıngan zaman yönetim mekanizması, Mİ sayısı 16'yı geçtikten sonra Amdahl yasasının olumsuz etkisine yakalanmış; iyimser zaman yönetim mekanizması ise bu etkiye Mİ sayısı 64'ü geçtikten sonra yakalanmıştır. Deneyler arasında en iyi sonuç 64 Mİ'nin iyimser zaman yönetim mekanizması ile kullanıldığı durumda ortaya çıkmış ve 11,23'lük bir hızlanma değerine ulaşılmıştır.

360×360'lık ızgarada Mİ sayısı düşük olduğunda yine sakıngan zaman yönetim mekanizmasının daha iyi bir başarım gösterdiği görülmüştür. Ancak yine Mİ sayısı arttığında iyimser zaman yönetim mekanizması sakıngan zaman yönetim mekanizmasını yakalamıştır. Ancak bu yakalama, 120×120'lik ızgaradaki deneylere

göre daha fazla Mİ kullanıldığında gerçekleşmiştir. Bu kez, iyimser zaman yönetim mekanizması 81 ve daha fazla Mİ kullanıldığında sakıngan zaman yönetim mekanizmasını geçmeyi başarmıştır. Sakıngan zaman yönetim mekanizması yine Amdahl yasasının olumsuz etkisine yakalanmış; iyimser zaman yönetim mekanizmasında ise 144 Mİ'ye kadar bu etki gözlenmemiştir. En iyi başarım 144 Mİ'nin yine iyimser zaman yönetim mekanizması ile kullanıldığı durumda gerçekleşmiş olup hızlanma değeri 12,36 olarak ölçülmüştür.

Kaynakların elvermemesinden dolayı 120×120'lik ızgarada en fazla 225 Mİ; 360×360'lık ızgarada ise en fazla 144 Mİ kullanılabilmiştir. Deneyler sonucunda iyimser zaman yönetim mekanizmasının etmen tabanlı benzetimlerde sakıngan zaman yönetim mekanizmasından daha iyi sonuçlar verebildiği gözlenmiştir. Özellikle Mİ sayısının artmasının iyimser zaman yönetim mekanizmasının başarımına daha olumlu şekilde etki ettiği gözlenmiştir. Bunun en belirgin nedeni, iyimser zaman yönetim mekanizmasının zaman uyumlama mekanizmalarını gevşetmesi sonucu koşutluğu daha da artırabilmesidir. Bu sonuçlar göstermektedir ki; iyimser zaman yönetim mekanizması etmen tabanlı benzetimler için daha ölçeklenebilir bir şekilde çalışabilmektedir.

Repast HPC yazılımının Zaman Bükülmesi mekanizmasıyla genişletilmesi ile genel amaçlı olarak geliştirilmiş bir modelleme ve benzetim yazılımının iyimser bir zaman yönetimi algoritmasıyla çalışabilmesi sağlanmış ve benzetimlerin daha kısa sürelerde çalıştırılabileceği gösterilmiştir. Her iki zaman yönetim mekanizmasının başarımının Repast HPC üzerinden etmen tabanlı benzetimlerdeki karşılaştırılmasına yönelik yapılan bu çalışmalar [123]'te sunulmuştur.

6. ZAMAN BÜKÜLMESİ MEKANİZMASININ İYİLEŞTİRİLMESİ

Zaman Bükülmesi mekanizmasındaki en büyük maliyetlerden bir tanesi geri sarma mekanizması çalıştırıldığında ortaya çıkar. Şekil 3.1'de görüldüğü gibi bir Mİ mızıkçı bir ileti aldığında öncelikle kendi durumunu daha önceden kaydedilmiş güvenli bir duruma çeker. Ardından mızıkçı iletinin zamanına kadar olan olayları yeniden işletir (ileri sarma). Mızıkçı iletiyi işlettikten sonra da benzetimi koşturmaya devam eder. Bu işlemler sırasında, eğer geri sarma aşamasından önce diğer Mİ'lere gönderilen iletilerden birinin düzeltilmesi veya yok edilmesi gerektiği fark edilirse bu hatayı telafi etmek üzere bir de karşıt ileti gönderilir. Tüm bu işlemlerden dolayı geri sarma mekanizması ciddi bir maliyet etkenidir. Zaman Bükülmesi mekanizmasında bir geri sarma işleminin maliyeti (2) eşitliğinde verilmiştir. Bu eşitlikte; C_R toplam geri sarma maliyetini, C_{SR} benzetim durumunun geri yüklenmesi aşamasının maliyetini, C_{CF} ileri sarma aşamasının maliyetini, C_{RE} yeniden oynatma aşamasının maliyetini ve C_{AM} ise hatalı iletilerin tespit edilmesi ve karşıt iletilerin hazırlanması aşamasının maliyetlerini temsil etmektedir.

$$C_R = C_{SR} + C_{CF} + C_{RE} + C_{AM} \quad (2)$$

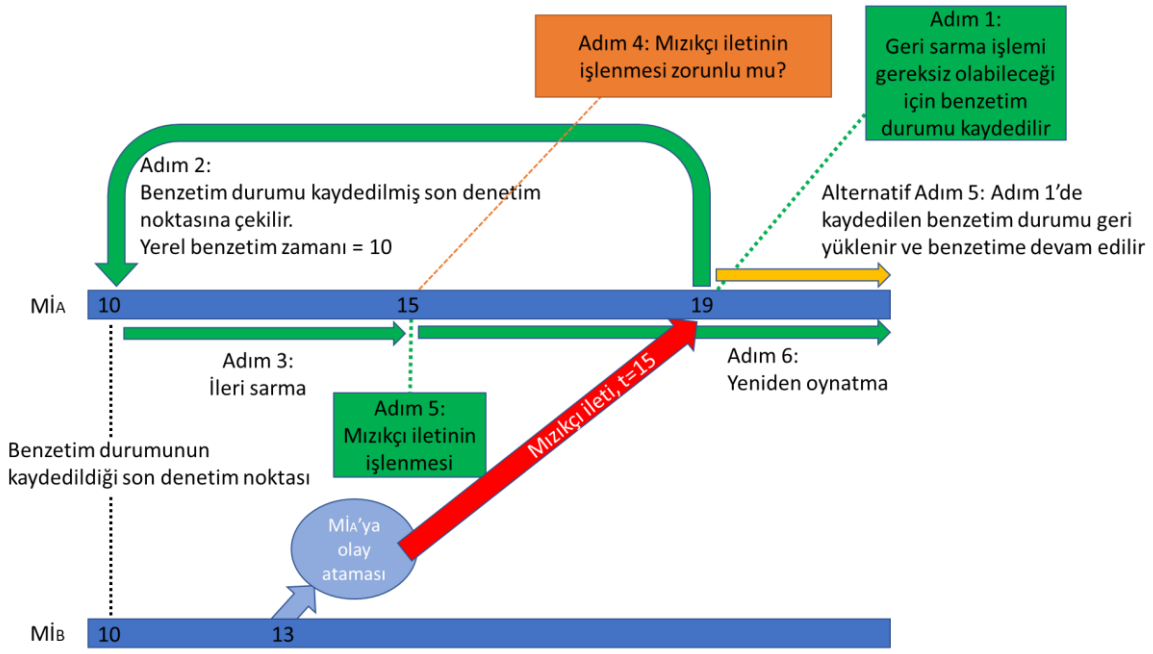
Zaman Bükülmesi mekanizmasının mızıkçı bir iletiyi alır almaz bir geri sarma işlemi başlatması bazı durumlarda gereksiz olabilir. Bu durumlarda gereksiz bir geri sarma işlemi yapılmakta ve gereksiz bir maliyet ortaya çıkmaktadır. Eğer mızıkçı iletinin gerçekten bir geri sarma gerektirip gerektirmediği tespit edilebilirse bu gereksiz maliyetten kurtulmak da mümkün olacaktır.

Bu tez çalışmasında önerilen yöntemler ile Zaman Bükülmesi mekanizmasını kullanan Mİ'lerin gereksiz geri sarma işlemlerini tespit edebilmesi ve önemli bir maliyetten kurtulması hedeflenmiştir. Önerilen yöntemlerden ilki "alt-durum kaydetme mekanizması" olarak adlandırılmış olup benzetim durumlarının sadece küçük bir kısmını kaydederek mızıkçı iletilerin geri sarmaya neden olup olmayacağını tespit etmeye çalışır. İkinci yöntem ise "Öngörülmüş Geri Sarma (ÖGS)" mekanizması olarak adlandırılmış olup, benzetim sırasında ileride alınabilecek mızıkçı iletilere karşı önlem almayı amaçlamaktadır. ÖGS

mekanizması bu tezin en önemli katkısını oluşturduğu için ayrı bir başlıkta daha ayrıntılı bir biçimde ele alınmıştır. Alt-durum kaydetme ve ÖGS mekanizmalarının gerekliliğinin daha kolay anlaşılması için, öncesinde alternatif bir başka yöntem olan durum farkı hesaplama mekanizması anlatılmıştır. Ancak bu mekanizma Zaman Bükülmesi mekanizmasında bir iyileştirme sağlamamaktadır.

6.1. Durum Farkı Hesaplama Mekanizması

Mızıkçı iletilerin benzetime etkisinin tespit edilebilmesi için benzetimin geçmiş zamandaki durumunun bilinmesine ihtiyaç vardır. Böylece mızıkçı iletinin alındığı zamanki benzetim durumunun mızıkçı iletiden etkilenip etkilenmediği tespit edilebilir. Eğer bir sürekli denetim noktası alma yaklaşımı kullanılıyorsa, mızıkçı iletiyi alan Mİ, mızıkçı iletinin sahip olduğu zaman damgasında kaydedilmiş olan geçmiş durumuna geri dönebilir. Böylece mızıkçı iletinin gerekliliği o anki benzetim durumu üzerinden incelenebilir. Ancak Bölüm 3.3'te anlatıldığı üzere sık denetim noktası alma yaklaşımlarının benzetimlerin koşum zamanına olan olumsuz etkisinden dolayı bu yaklaşımlar yerine değişken aralıklı seyrek denetim noktası yaklaşımları tercih edilmektedir. Bu durumda da mızıkçı iletinin etkisinin anlaşılabilmesi için Mİ'nin mızıkçı iletinin zamanına geri dönmesi gerekmektedir. Şekil 6.1'de bu gereklilik özetlenmiştir.



Şekil 6.1. Geri sarma işleminin gerekliliğine karar verilmesi

Şekil 6.1'e göre mızıkçı iletinin geri sarma işlemine neden olup olmayacağının anlaşılabilmesi için zaten geri sarma işlemlerinin bir kısmı başlatılmak zorundadır. Mızıkçı ileti alan bir Mİ öncelikle o anki yerel durumunun bir kopyasını kaydetmelidir (adım 1). Adım 1'in neden gerektiği alternatif adım 5'te anlatılacaktır. Adım 2 ve adım 3, Zaman Bükülmesi mekanizmasında olduğu gibi Mİ'nin durumunu bir önceki denetim noktasına çekmesini ve ardından ileri sarma işlemi uygulamasını sağlar. Böylece Mİ'nin yerel zamanı mızıkçı iletinin işlenmesi gereken zamana gelmiş olur. Bu noktada Mİ, mızıkçı iletinin gerçekten benzetime etki edip etmeyeceğini tespit edebilir (adım 4). Bunun için literatürde de örnekleri bulunan durum farkı hesaplama veya durum eşleştirme yöntemleri (örneğin [124]) kullanılarak mızıkçı ileti işlendiğinde ve işlenmediğinde ortaya çıkacak iki durumun karşılaştırılması sağlanır. Eğer gerçekten mızıkçı iletinin işlenmesi gerektiği sonucuna varılırsa mızıkçı ileti işleme alınır (adım 5) ve ardından yeniden oynatma aşaması (adım 6) ve karşıt iletilerin tespit edilmesi aşamaları işletilir. Ancak mızıkçı iletinin işletilmesinin gerekli olmadığı anlaşılırsa, alternatif adım 5 uygulanarak benzetim durumu önce adım 1'de kaydedilmiş olan benzetim durumuna çekilir; ardından da benzetim kaldığı yerden devam ettirilir.

Bu yöntemdeki en belirgin dezavantaj, yöntemin getirdiği ek iş yüküdür. Zaman Bükülmesi mekanizmasında zaten pahalı bir maliyet oluşturan benzetim durumunun denetim noktasına geri yüklenmesi ve ileri sarma işlemleri bu yöntemde de mevcuttur. Ek olarak, Şekil 6.1'de gösterilen adım 1'de gerçekleşen durum kaydetme, adım 4'te gerçekleşen durum farkı hesaplama ve alternatif adım 5'te gerçekleşen durum geri yükleme işlemlerinin maliyetleri de bu yöntemde söz konusudur. Dolayısıyla geri sarmalardan kaçınmak için durum farkı hesaplama yöntemlerinden yararlanılmak istenirse ortaya çıkacak toplam maliyet (3) eşitliğinde gösterilmiştir. Bu eşitlikte; C_R toplam geri sarma maliyetini, C_{SS} durum kaydetme aşamasının maliyetini, C_{SR} benzetim durumunun geri yüklenmesi aşamasının maliyetini, C_{CF} ileri sarma aşamasının maliyetini, C_{SD} durum farkı hesaplama aşamasının maliyetini, C_{RE} yeniden oynatma aşamasının maliyetini, C_{AM} hatalı iletilerin tespit edilmesi ve karşıt iletilerin hazırlanması aşamasının maliyetlerini temsil etmektedir.

$$C_R = C_{SS} + C_{SR} + C_{CF} + C_{SD} + \begin{cases} C_{RE} + C_{AM}, & geri\ sarma\ kaçınılmazsa \\ C_{SR}, & geri\ sarma\ gereksizse \end{cases} \quad (3)$$

(2) ve (3) eşitlikleri göz önüne alınarak Zaman Bükülmesi mekanizması ve durum farkı hesaplama mekanizmasıyla genişletilmiş algoritmanın genel hatlarıyla karşılaştırılması şu şekilde değerlendirilebilir:

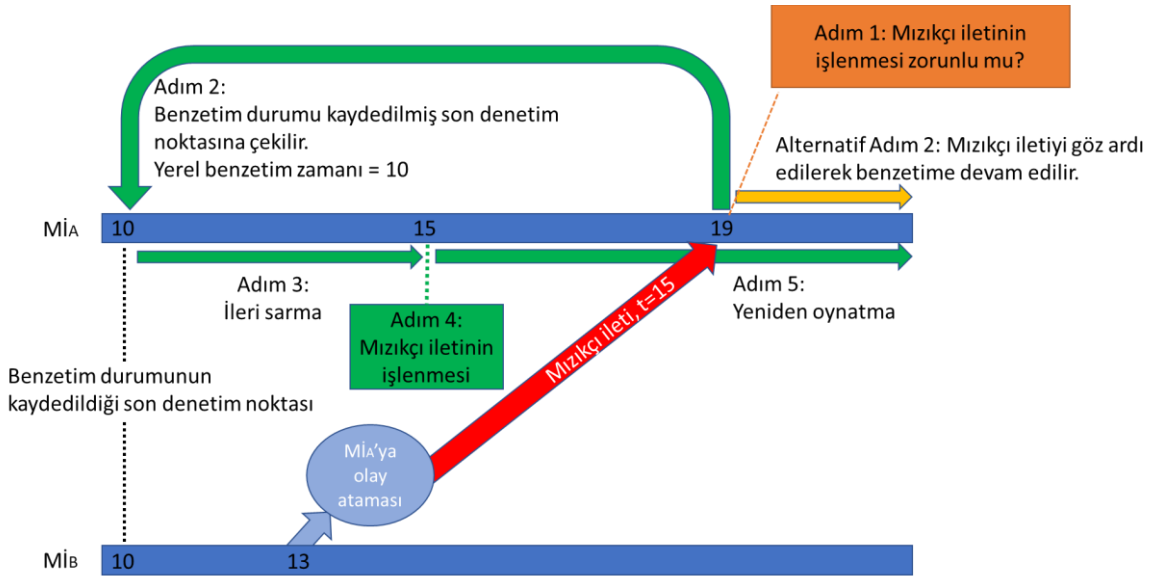
- Eğer bir geri sarma işlemi kaçınılmazsa durum farkı yöntemi Zaman Bükülmesi mekanizmasından $C_{SS} + C_{SD}$ kadar daha pahalı olacaktır.
- Eğer bir geri sarma işleminin gereksiz olduğu tespit edilirse durum farkı hesaplama mekanizmasının $C_{SS} + C_{SD} + C_{SR}$ maliyeti ve Zaman Bükülmesi mekanizmasının $C_{RE} + C_{AM}$ maliyeti karşılaştırılmalıdır. Ancak durum kaydetme (C_{SS}) ve durum geri yükleme (C_{SR}) işlemleri genellikle oldukça pahalı yöntemler olduğu için [43] durum farkı hesaplama mekanizmasının Zaman Bükülmesi mekanizmasına göre daha pahalı olma ihtimali ortaya çıkmaktadır. Eğer yeniden oynatma adımı yeterince uzun bir benzetim zamanı boyunca gerçekleşmeyecekse ve buna bağlı olarak yüksek bir maliyete sahip olmayacaksa Zaman Bükülmesi mekanizması daha ucuz olacaktır. C_{SS} ve C_{SR} maliyetleri görmezden gelinse bile, ancak ve ancak durum farkını hesaplayan yöntemin maliyetinin de (C_{SD}) çok düşük olması durumunda Zaman Bükülmesi mekanizmasından daha ucuz bir maliyet ortaya çıkacaktır.

Özetle, durum farkı hesaplama yönteminin birçok durum için gerçekleştirilmeye değmeyeceği ve Zaman Bükülmesi mekanizmasının birçok durumda daha iyi olacağı değerlendirilmiştir. Ayrıca [33] çalışmasında da benzer ek maliyetlerle karşılaşıldığı görülmüştür. Bu durum, Zaman Bükülmesi mekanizmasının iyileştirilmesi için daha az maliyetli bir çözüme ihtiyaç olduğu sonucunu ortaya çıkarmıştır.

6.2. Alt-Durum Kaydetme Mekanizması

Durum farkı hesaplama yönteminin olumsuzluğu mızıkçı iletilerin geri sarmaya neden olup olmayacağını geçmiş bir duruma döndükten sonra anlayabilmesinden kaynaklanmaktadır. Bunun yerine Mİ'lerin, mızıkçı iletileri alır almaz inceleyebilmesi durumunda bu olumsuzluğun üstesinden gelinebilir. Ancak Mİ'ler geçmiş

durumlarının tamamına sahip değildirler; yalnızca son denetim noktasında kaydedilmiş olan geçmiş durumlarına sahiptirler. Bu yüzden mızıkçı iletinin benzetime etki edip edemeyeceğini bilmek Mİ için mümkün olmayabilir. Bu sorunları aşmak için Zaman Bükülmesi mekanizmasına bir eklenti olarak önerdiğimiz alt-durum kaydetme mekanizması Mİ'lerin, benzetim durumlarının yalnızca belli bir alt kümesini kaydetmesini önerir. Kaydedilen bu alt kümeler, bu tez çalışmasının kalan kısmında alt-durum adıyla anılacaktır. Alt durumlar, Mİ'ler tarafından benzetim durumu her değiştiğinde kaydedilir. Bu alt-durumlar, ileride bir mızıkçı ileti alınması durumunda mızıkçı iletinin gerçekten bir geri sarmaya neden olup olmayacağının incelenmesine olanak sağlar. Bu sayede Mİ'ler, mızıkçı ileti alınır alınmaz acil bir geri sarma işlemi başlatmak zorunda kalmaz. Önerilen yöntem, Şekil 6.2'de verilen akışın gerçekleştirilmesini sağlar.



Şekil 6.2. Mızıkçı iletinin geri sarma işlemi başlamadan önce incelenebildiği yöntemlerde izlenmesi gereken akış

Alt-durum kaydetme mekanizması, durum farkı hesaplama mekanizmasından farklı olarak adım 1'de mızıkçı iletiyi inceler. Bu inceleme sonucunda Mİ bir geri sarma yapıp yapmayacağı kararını verir. Eğer geri sarma kaçınılmazsa, Şekil 6.2'de gösterilen adım 2, 3, 4 ve 5 gerçekleştirilerek ve gerekli olan karşıt-iletiler gönderilerek Zaman Bükülmesi mekanizmasındaki geleneksel geri sarma işlemi yapılır. Ancak; adım 1'de mızıkçı iletinin benzetim sonucunu değiştirmeyecek bir ileti

olduğu tespit edilirse geri sarma işlemine başlanmaz ve benzetim alternatif adım 2'de gösterildiği üzere kaldığı yerden devam eder.

Alt-durum kaydetme mekanizması, her durum değişmesinde alt-durumların kaydedilmesini gerektirdiği için ek bir maliyetle ortaya çıkmaktadır. Alt-durumlar benzetimin tüm durumuna göre oldukça küçük boyutlara sahiptir. Bölüm 6.2.2'de, tüm durumların kaydedilmesi ve alt-durumların kaydedilmesi arasındaki maliyet farkı bir durum çalışması üzerinden örneklendirilmiştir. Alt-durum kaydetme mekanizmasındaki bir geri sarma işlemi maliyeti ise (4) eşitliğinde verilmiştir. Bu eşitlikte; C_R toplam geri sarma maliyetini, C_{SMI} mızıkçı iletinin incelenmesi aşamasının maliyetini, C_{SR} benzetim durumunun geri yüklenmesi aşamasının maliyetini, C_{CF} ileri sarma aşamasının maliyetini, C_{RE} yeniden oynatma aşamasının maliyetini ve C_{AM} hatalı iletilerin tespit edilmesi ve karşıt iletilerin hazırlanması aşamasının maliyetlerini temsil etmektedir.

$$C_R = C_{SMI} + \begin{cases} C_{SR} + C_{CF} + C_{RE} + C_{AM}, & geri\ sarma\ kaçınılmazsa \\ 0, & geri\ sarma\ gereksizse \end{cases} \quad (4)$$

(2) ve (4) eşitlikleri göz önüne alınarak Zaman Bükülmesi mekanizması ve alt-durum kaydetme mekanizmasıyla genişletilmiş algoritmanın genel hatlarıyla karşılaştırılması şu şekilde değerlendirilebilir:

- Eğer bir geri sarma işlemi kaçınılmazsa alt-durum kaydetme yöntemi Zaman Bükülmesi mekanizmasından C_{SMI} kadar daha pahalı olacaktır.
- Eğer bir geri sarma işleminin gereksiz olduğu tespit edilirse alt-durum kaydetme mekanizması C_{SMI} kadar bir maliyete sahip olacakken, Zaman Bükülmesi mekanizması $C_{SR} + C_{CF} + C_{RE} + C_{AM}$ kadarlık bir maliyete neden olacaktır. Bu durumda mızıkçı iletinin incelenmesi için maliyet açısından ucuz bir yöntem kullanılırsa, alt-durum kaydetme mekanizması Zaman Bükülmesi mekanizmasına göre işlem yükü açısından daha az maliyetli olacaktır.

6.2.1. Alt-Durumların Kaydedilmesi ve Mızıkçı İletilerin İncelenmesi

Alt-durumları kaydetme mekanizması Bölüm 5'te ayrıntıları verilen RHPC_TW yazılımı genişletilerek gerçekleştirilmiştir. Önerilen mekanizma, daha iyi anlaşılabilmesi için Civil Violence [120] modeli üzerinden örneklenmiştir. Alt-durum

kaydetme mekanizması benzetimi yapılacak modelin etkileşim modelinden yararlanır. Etkileşim modeli ile kast edilen; benzetimi yapılacak modelin anlamsal özelliklerini (İng. *semantics*) kullanarak etmenlerin davranışının hangi etmenler tarafından hangi koşullar altında etkileneceğini bildiren bir belirtimdir. Örneğin; Bölüm 5.5'te ayrıntıları verilen modeldeki kurallar gereği, bu modelin etkileşim modeli şu şekilde özetlenebilir:

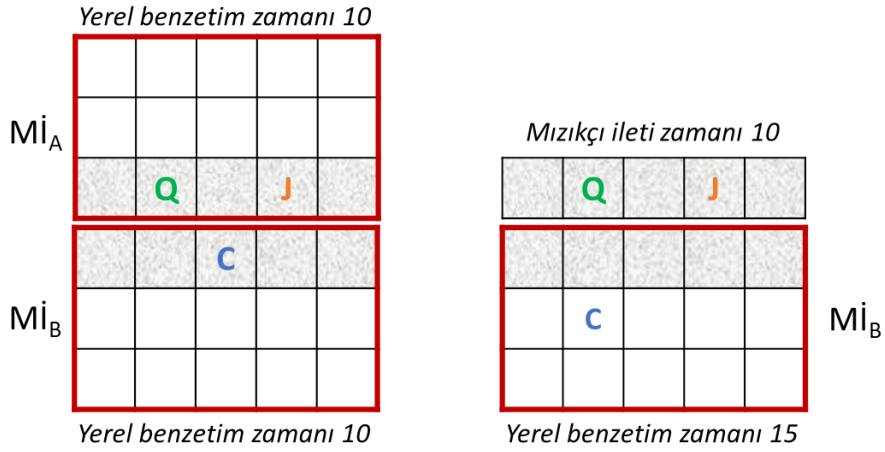
- Bir polis etmeni (C), etrafındaki eylemci etmenlerle (A) etkileşime girer. Bir başka deyişle, polis etmenlerinin davranışı etrafında bulunan eylemci etmenlere göre değişiklik gösterebilir.
- Bir sakin etmen (Q), etrafındaki polis ve eylemci etmenlerle etkileşim halindedir.
- Bir eylemci etmen, etrafındaki polis etmenlerle etkileşim halindedir.
- Bir mahkum etmen (J), etrafındaki hiçbir etmenle etkileşim halinde değildir.

Bu etkileşim modeline göre aşağıdaki durumlar gerçekleştiğinde herhangi bir geri sarma işlemine gerek olmayacaktır:

- Bir polis etmen etrafındaki polis, sakin ve mahkum etmenleri göremediğinde,
- Bir sakin etmen etrafındaki sakin ve mahkum etmenleri göremediğinde,
- Bir eylemci etmen, etrafındaki eylemci, sakin ve mahkum etmenleri göremediğinde,
- Bir mahkum etmen etrafındaki herhangi bir etmeni göremediğinde.

Alt durum kaydetme mekanizmasının kaçınmaya çalıştığı geri sarma işlemleri Bölüm 5.3.1'de bahsedilen "etmenlerin komşu Mİ'lere görünürlüğü" türünden olaylardır. Diğer türdeki olaylar için geri sarma işlemlerinden kaçınma yapılmaz. Şekil 6.3'te geri sarmaya neden olmayacak bir durum örneklenmiştir. Bu örnekte ilk olarak 10. zaman adımında her iki Mİ'nin durumu gösterilmiş olup, komşuluk alanının 1 hücre olduğu varsayılmıştır. Yani etmenler, etrafındaki 8 hücrede yer alan etmenlerle etkileşime girebilmektedir. Bu örnekteki duruma bakıldığında, t anında $Mİ_A$'nın tampon alanında birer adet Q ve J türünden etmen bulundurduğu; bu sırada $Mİ_B$'nin ise tampon alanında C türünden bir etmen bulundurduğu görülmektedir. Her iki Mİ, Zaman Bükülmesi mekanizmasının ilkeleri gereği yerel zamanlarını birbirleriyle zaman uyumlama yapmadan ilerletebilirler. Bu örnekte $Mİ_B$'nin, $Mİ_A$'nın

Q ve J etmenlerini bildirdiği iletiyi 15. zaman adımında aldığı varsayılsın. Alınan bu iletinin zaman adımı 10 olduğu için Zaman Bükülmesi mekanizması bu iletiyi mızıkçı bir ileti olarak tanımlayacak ve bir geri sarma işlemi başlatacaktı. Ancak alt-durum kaydetme mekanizması, geri sarmayı başlatmadan önce bu iletinin içeriğini inceleyerek geri sarmadan kaçınabilir. Bunun için sahip olması gereken şey, 10. zaman adımındaki kendi geçmiş durumudur. Ancak geçmiş durumunun tamamını bilmesine gerek yoktur. Bunun yerine yalnızca tampon alandaki etmenlerin durumlarını bilmesi yeterli olacaktır. Alt-durum kaydetme mekanizması $Mİ_B$'nin şu soruyu sormasını sağlamaktadır: "Eğer bu mızıkçı ileti 10. zaman adımında sorunsuz bir şekilde alınmış olsaydı, C etmeninin davranışına bir etkide bulunacak mıydı?". Örneklenen bu modelde C etmeninin davranışı etrafındaki Q ve J etmenlerine göre değişmeyeceği için $Mİ_B$ geri sarma işlemi yapmak zorunda olmadığını anlayacaktır. Ancak; örneğin Q etmeninin yerinde bir A etmeni bulunsaydı, mızıkçı iletinin zamanında işlenmiş olması benzetimin durumunu değiştirmesine neden olabilirdi. Dolayısıyla $Mİ_B$ geri sarma işleminin kaçınılmaz olduğunu anlayacak ve Zaman Bükülmesi mekanizmasında olduğu gibi, geri sarma işlemini başlatacaktı.



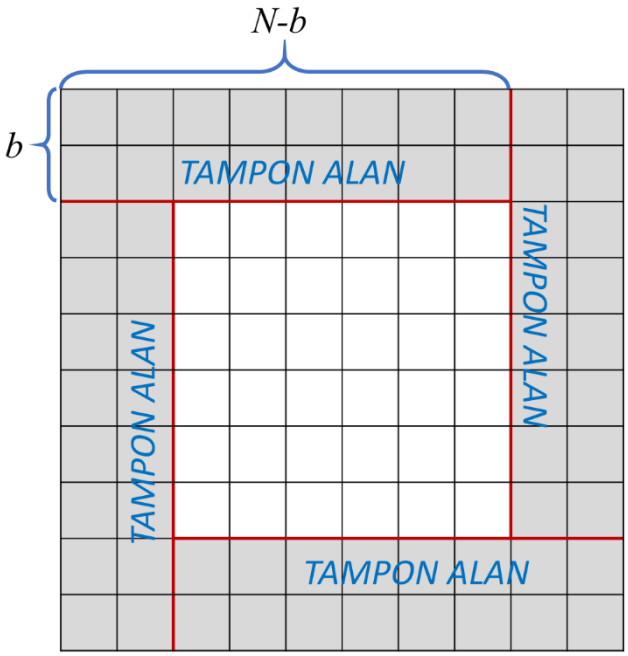
Şekil 6.3. Geri sarmaya neden olmayacak bir mızıkçı ileti örneği

6.2.2. Alt-Durum Kaydetme Mekanizmasının Getirdiği Ek İş Yükü

Alt-durum kaydetme mekanizması benzetimlere ek bir iş yükü getirmektedir. Bu maliyet benzetimin tüm durumunu kaydetmeye göre oldukça ucuzdur. Örneğin, her

bir Mİ'nin NxN'lik bir ızgara ortamında A adet etmeni yönettiği ve komşuluk alanı büyüklüğünün b olduğu bir benzetimde;

- Eğer tüm benzetim durumu kaydedilecek olsaydı, A adet etmenin tamamı tüm öznitelikleriyle birlikte kaydedilecekti. Örneğin; Civil Violence modelinde her bir etmenin 6 tam sayı (`id`, `startingRank`, `agentType`, `currentRank`, `score`, `jailTerm`), 4 de gerçek sayı (`riskAversion`, `perceivedHardship`, `grievance`, `arrestProbability`) olmak üzere toplamda 10 özniteliği bulunmaktadır. Ayrıca her bir etmen için Repast HPC 6 tam sayı kullanarak konum bilgisini tutmaktadır. Bu 6 tam sayının 4'ü AgentId bilgisini (bkz. Ek - 3), 2'si ise iki boyutlu uzaydaki konum bilgisini temsil etmektedir. Bu durumda benzetim durumunu kaydetmek için her bir etmen başına toplam 12 tam sayı, 4 de gerçek sayı kaydedilmelidir. Geliştirilen benzetimler GNU Compiler Collection (GCC) derleyicisi kullanılarak derlendiği için bir tam sayı 4 byte'lık, bir gerçek sayı ise 8 byte'lık bir bellek alanına kaydedilmektedir [125]. Bu durumda en az $A \cdot (12 \cdot 4 + 4 \cdot 8) = 80 \cdot A$ kadarlık bir bellek alanına ihtiyaç duyulacaktır.
- Alt-durum kaydetme mekanizması yalnızca tampon alandaki etmenlerin kaydedilmesini gerektirir. Tampon alanlarda bulunan toplam hücre sayısı Şekil 6.4'te görülebileceği gibi $4 \cdot b \cdot (N - b)$'dir. Buna göre bir t anında, tampon alanlarda bulunan etmen sayısı ortalama olarak $A \cdot \frac{4 \cdot b \cdot (N-b)}{N^2}$ olacaktır. Yine Civil Violence modeli ele alındığında, etmenlerin tüm öznitelikleri kaydedilmek istenirse, toplamda $80 \cdot A \cdot \frac{4 \cdot b \cdot (N-b)}{N^2}$ byte kadar bir bellek alanına ihtiyaç olduğu görülmektedir. Alt-durum kaydetme mekanizması tüm özniteliklerin kaydedilmesini gerektirmez. Örneğin; Civil Violence modelinde etmenlerin konum bilgilerini tutan iki adet gerçek sayı ve etmen türlerini tutan bir adet tam sayının kaydedilmesi yeterlidir. Bu durumda toplamda $20.A.\frac{4.b.(N-b)}{N^2}$ byte kadar bir bellek alanına ihtiyaç vardır.



Şekil 6.4. Her bir Mİ'nin NxN'lik bir ızgara ortamını yönettiği ve komşuluk alanı büyüklüğünün b olduğu bir benzetimin görünümü

Yukarıda anlatılan bellek alanı ihtiyaçlarını somut bir örnek üzerinden karşılaştırmak gerekirse; 1000 etmenin kullanıldığı, her bir Mİ'nin 1000×1000'lik bir ızgarayı yönettiği ve komşuluk alanı büyüklüğünün 1 olduğu bir durumda;

- Tüm durumlar kaydedilirse her bir zaman adımında yaklaşık 80000 byte'lık bir bellek alanına ihtiyaç olduğu görülmektedir.
- Alt-durum kaydetme mekanizması ise her bir zaman adımında yaklaşık olarak ortalama 80 byte'lık bir bellek alanına ihtiyaç duyarak, geri sarmaların gerekli olup olmadığına karar verebilmektedir.

Bu maliyetlerin yanı sıra, alt-durum kaydetme mekanizmasında hangi etmenlerin kaydedileceğinin belirlenmesinin de bir maliyet getirebileceği düşünülebilir. Ancak, bu etmenlerin aranması için ek bir iş yükü gerekmemektedir. Çünkü bu etmenler komşu Mİ'lere gönderilmesi gereken etmenlerdir. Dolayısıyla alt-durum kaydetme mekanizması kullanılmasa bile bu etmenler zaten benzetim sırasında tespit edilmesi gereken etmenlerdir.

6.2.3. Alt-Durum Kaydetme Mekanizmasının Zaman Bükülmesi Mekanizmasıyla Karşılaştırılması

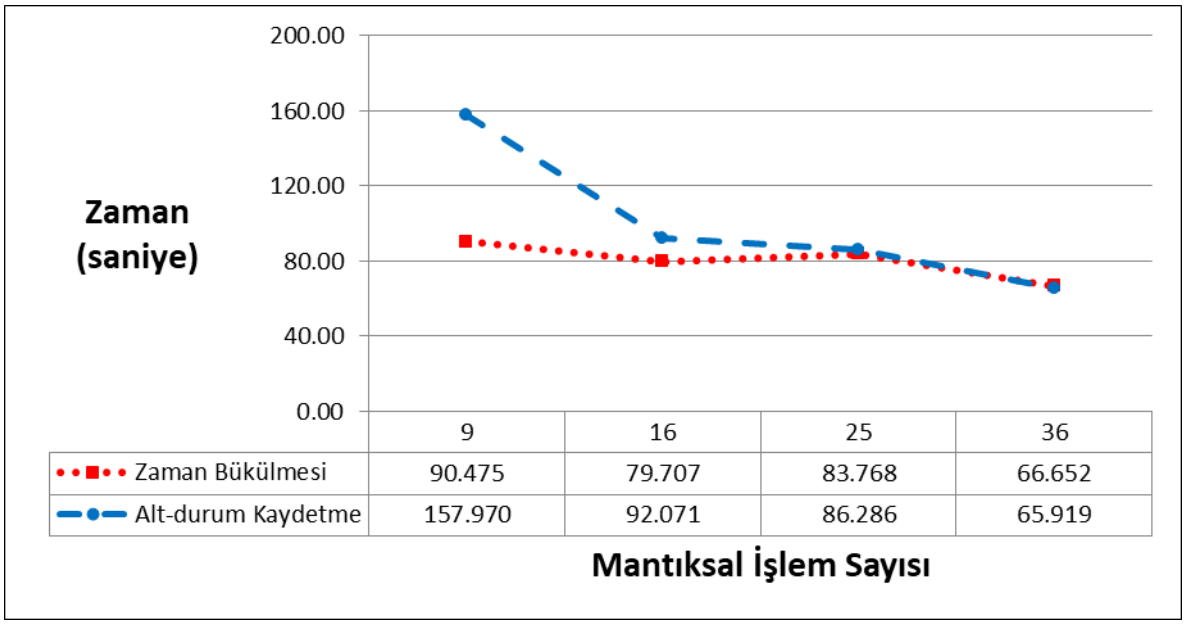
Alt-durum kaydetme mekanizmasının başarımını Zaman Bükülmesi mekanizmasıyla karşılaştırabilmek için her iki algoritmayı kullanan benzetimler bir YBH sisteminde denenmiştir. Bu YBH sistemi ODTÜ Bilgisayar Mühendisliği tarafından sunulan, Linux tabanlı ve Slurm [126] iş kuyruğu arayüzüne sahip olan bir sistemdir. Sistemde toplamda 189 GB bellek alanı ve her biri 16 çekirdekten oluşan 2.3 GHz frekansa sahip 4 adet AMD Opteron 6376 işlemci bulunmaktadır. Sistem aynı anda 40 çekirdeğin kullanımına izin verdiği için deneyler en çok 40 çekirdekle gerçekleştirilebilmiştir. Çekirdeklerin çoklu-iş parçacıklı (İng. *multi-threaded*) kullanılabilmesi özelliğinden yararlanılmamıştır.

Deneylerde Civil Violence modelinin benzetimi 2 boyutlu bir ızgara ortamı üzerinde 500 benzetim adımı boyunca gerçekleştirilmiştir. Bu model için komşuluk alanı 1, 2 ve 3 hücre; Mİ sayısı 9, 16, 25 ve 36; etmen sayısı ise 12960 ve 25920 olmak üzere farklı deneyler gerçekleştirilmiştir. Denetim noktası sıklıklarını belirlemek için Bölüm 5.3.2'de ayrıntıları verilen TAÇA algoritması kullanılmıştır.

Yapılan deneyler sonucunda her iki algoritmanın kullanımı durumunda gözlenen benzetim tamamlanma süreleri Çizelge 6.1 – Çizelge 6.6'da verilmiş ve bu veriler Şekil 6.5 – Şekil 6.10'da görselleştirilmiştir. Her bir deney en az 4 kez tekrarlanmış olup, bu şekil ve çizelgelerde bu deneylerin ortalama değerlerine ve standart sapmalarına yer verilmiştir. Ayrıca bu algoritmalar kullanıldığında elde edilen hızlanma değerleri ise Şekil 6.11 ve Şekil 6.12'de gösterilmiştir.

Çizelge 6.1. Etmen sayısı 12960 ve komşuluk alanı büyüklüğü 1 hücre iken benzetimin farklı sayılardaki Mİ'ler ile çalıştırılma süresi ve standart sapma değerleri

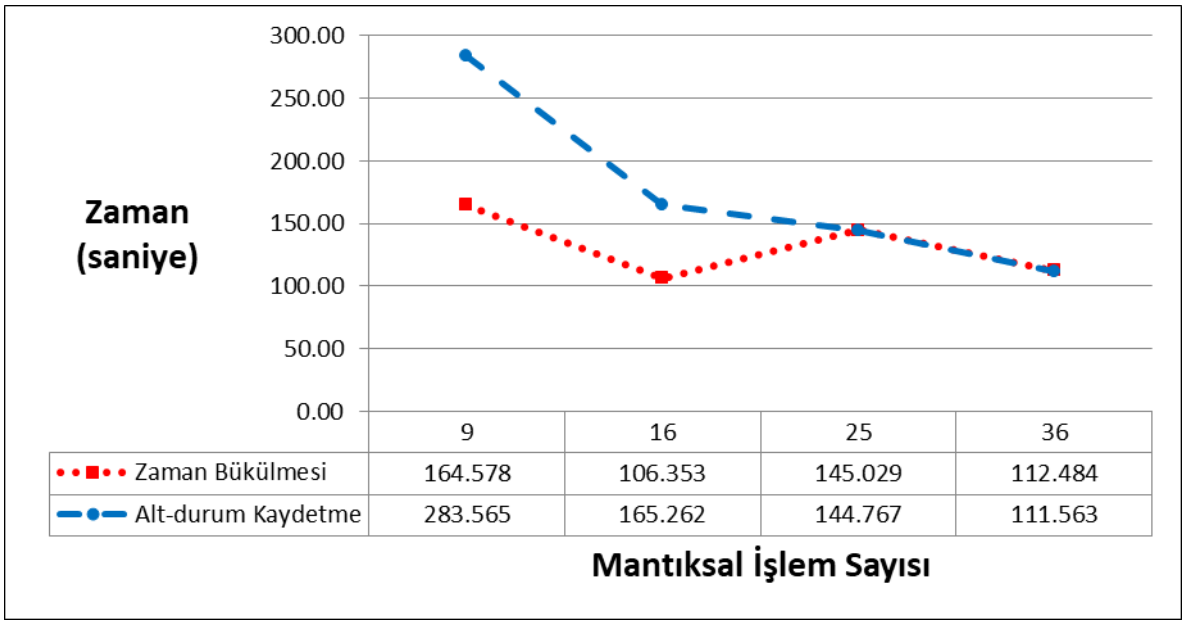
		Mİ sayısı			
		9	**16**	**25**	**36**
Zaman Bükülmesi	**Geçen zaman [s]**	90,48	79,71	83,77	66,65
	Standart sapma (σ) [s]	0,47	19,99	2,25	0,25
Alt-durum Kaydetme	**Geçen zaman [s]**	157,97	92,07	86,29	65,92
	Standart sapma (σ) [s]	4,07	4,25	1,27	0,14



Şekil 6.5. Etmen sayısı 12960 ve komşuluk alanı büyüklüğü 1 hücre iken benzetimin farklı sayılardaki Mİ'ler ile çalıştırılma süresi

Çizelge 6.2. Etmen sayısı 12960 ve komşuluk alanı büyüklüğü 2 hücre iken benzetimin farklı sayılardaki Mİ'ler ile çalıştırılma süresi ve standart sapma değerleri

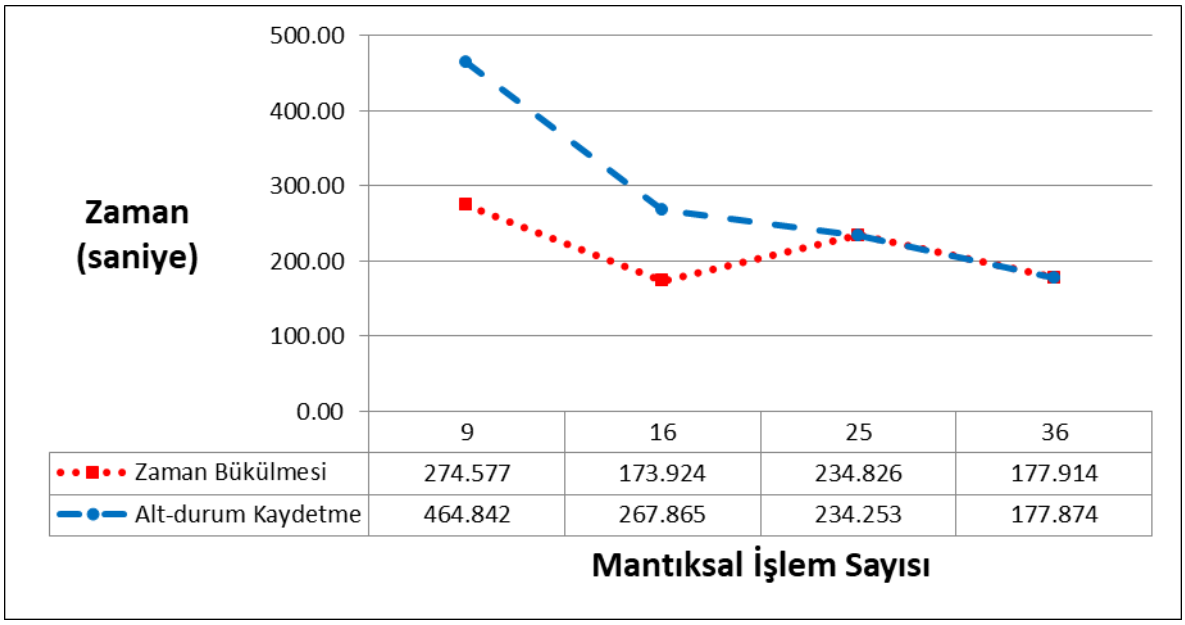
		Mİ sayısı			
		9	**16**	**25**	**36**
Zaman Bükülmesi	**Geçen zaman [s]**	164,58	106,35	145,03	112,48
	Standart sapma (σ) [s]	4,42	0,94	15,61	0,86
Alt-durum Kaydetme	**Geçen zaman [s]**	283,57	165,26	144,77	111,56
	Standart sapma (σ) [s]	5,49	7,50	2,41	0,71



Şekil 6.6. Etmen sayısı 12960 ve komşuluk alanı büyüklüğü 2 hücre iken benzetimin farklı sayılardaki Mİ'ler ile çalıştırılma süresi

Çizelge 6.3. Etmen sayısı 12960 ve komşuluk alanı büyüklüğü 3 hücre iken benzetimin farklı sayılardaki Mİ'ler ile çalıştırılma süresi ve standart sapma değerleri

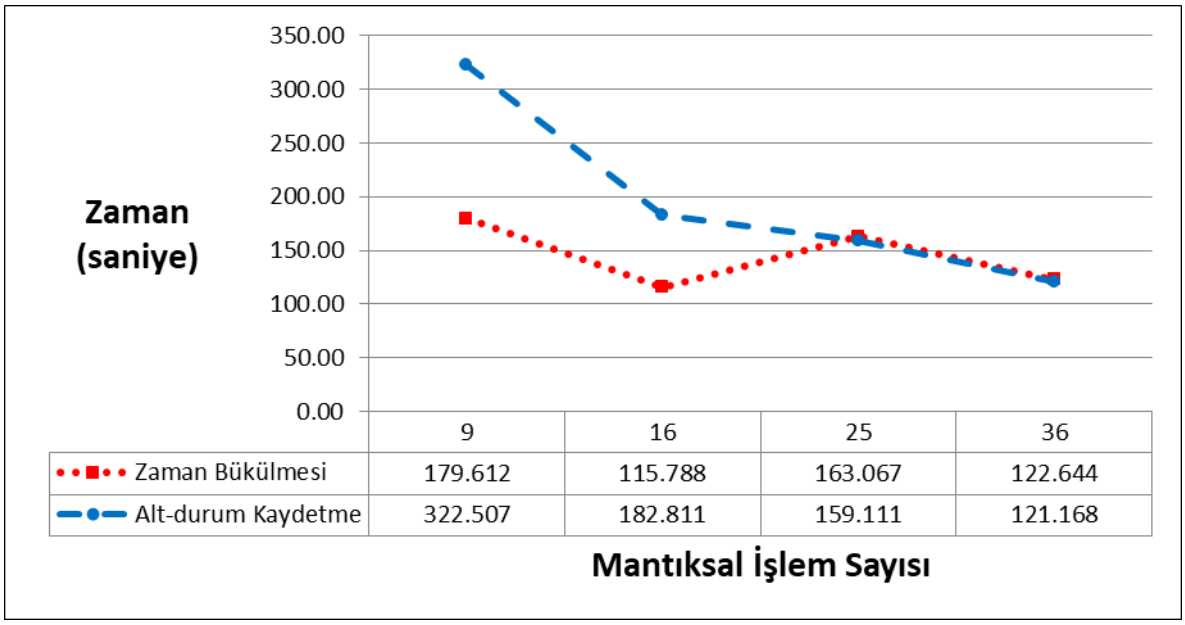
		Mİ sayısı			
		9	**16**	**25**	**36**
Zaman Bükülmesi	**Geçen zaman [s]**	274,58	173,92	234,83	177,91
	Standart sapma (σ) [s]	6,21	0,52	5,03	0,61
Alt-durum Kaydetme	**Geçen zaman [s]**	464,84	267,87	234,25	177,87
	Standart sapma (σ) [s]	15,33	4,29	4,52	0,70



Şekil 6.7. Etmen sayısı 12960 ve komşuluk alanı büyüklüğü 3 hücre iken benzetimin farklı sayılardaki Mİ'ler ile çalıştırılma süresi

Çizelge 6.4. Etmen sayısı 25920 ve komşuluk alanı büyüklüğü 1 hücre iken benzetimin farklı sayılardaki Mİ'ler ile çalıştırılma süresi ve standart sapma değerleri

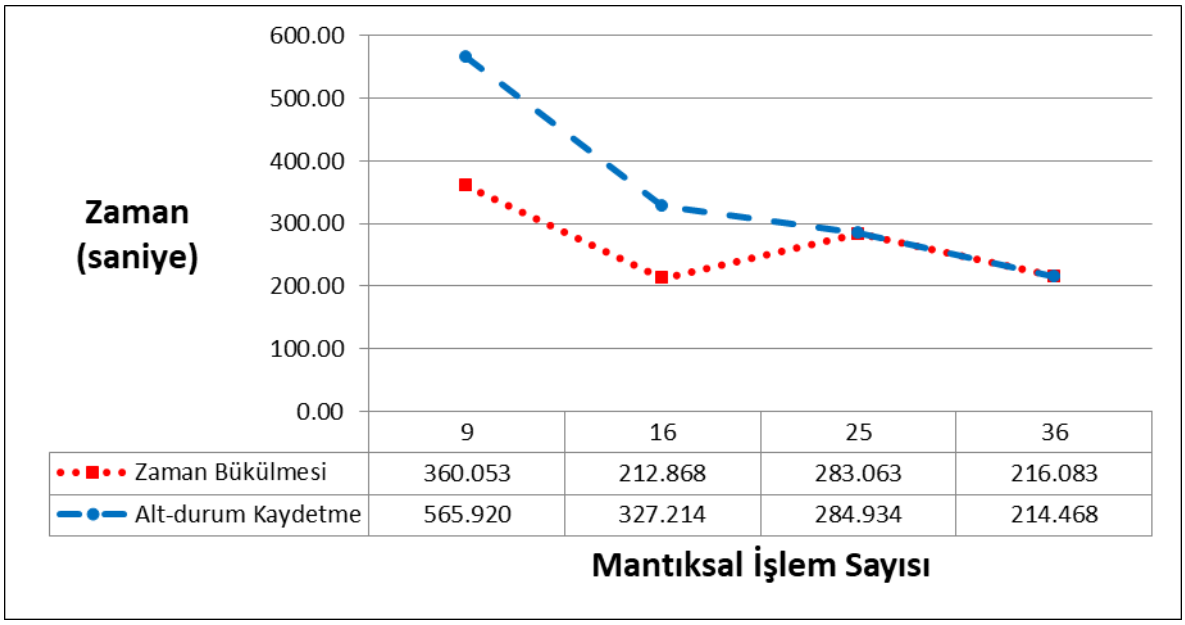
		Mİ sayısı			
		9	**16**	**25**	**36**
Zaman Bükülmesi	**Geçen zaman [s]**	179,61	115,79	163,07	122,64
	Standart sapma (σ) [s]	5,25	0,78	3,08	5,27
Alt-durum Kaydetme	**Geçen zaman [s]**	322,51	182,81	159,11	121,17
	Standart sapma (σ) [s]	12,29	8,43	5,95	0,99



Şekil 6.8. Etmen sayısı 25920 ve komşuluk alanı büyüklüğü 1 hücre iken benzetimin farklı sayılardaki Mİ'ler ile çalıştırılma süresi

Çizelge 6.5. Etmen sayısı 25920 ve komşuluk alanı büyüklüğü 2 hücre iken benzetimin farklı sayılardaki Mİ'ler ile çalıştırılma süresi ve standart sapma değerleri

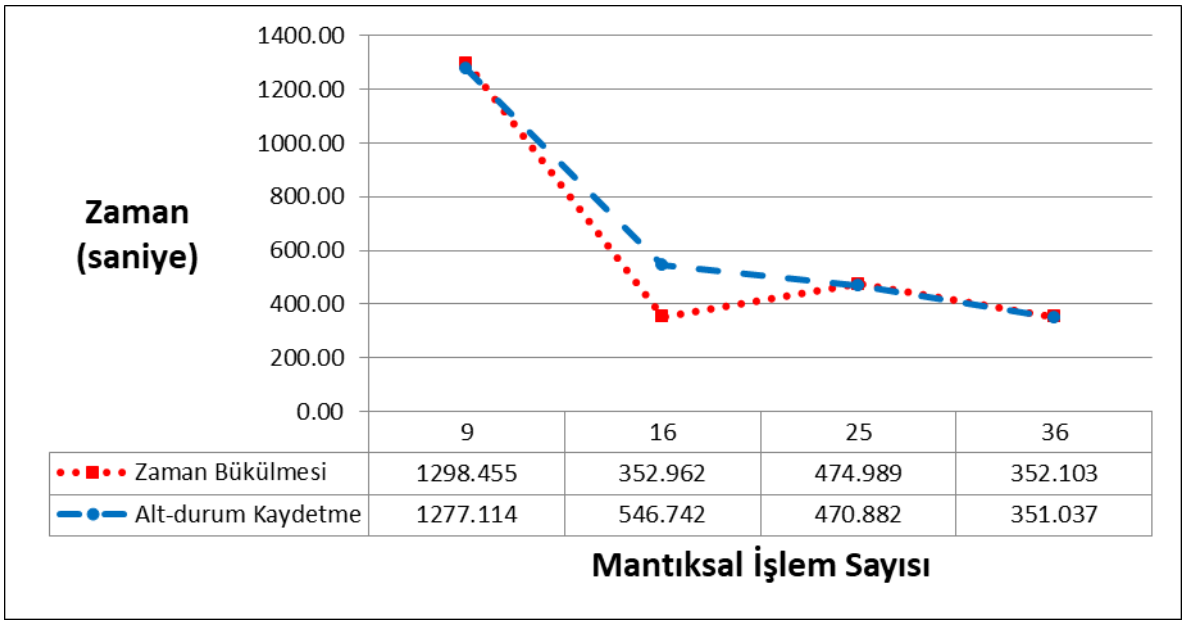
		Mİ sayısı			
		9	**16**	**25**	**36**
Zaman Bükülmesi	**Geçen zaman [s]**	360,05	212,87	283,06	216,08
	Standart sapma (σ) [s]	2,84	4,81	3,61	4,73
Alt-durum Kaydetme	**Geçen zaman [s]**	565,92	327,21	284,93	214,47
	Standart sapma (σ) [s]	19,87	5,43	8,53	0,81



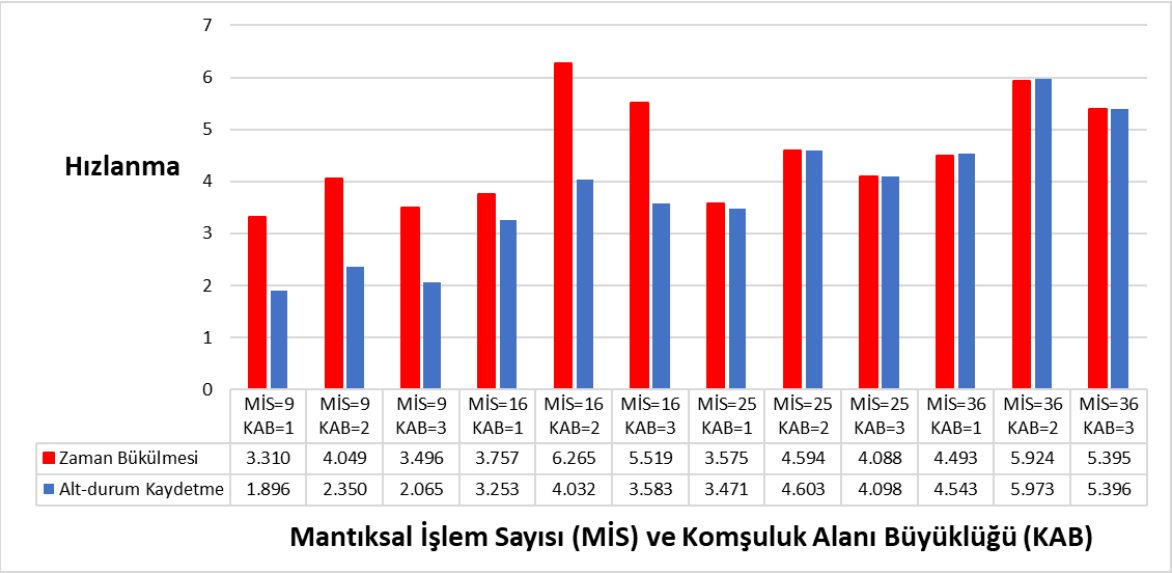
Şekil 6.9. Etmen sayısı 25920 ve komşuluk alanı büyüklüğü 2 hücre iken benzetimin farklı sayılardaki Mİ'ler ile çalıştırılma süresi

Çizelge 6.6. Etmen sayısı 25920 ve komşuluk alanı büyüklüğü 3 hücre iken benzetimin farklı sayılardaki Mİ'ler ile çalıştırılma süresi ve standart sapma değerleri

		Mİ sayısı			
		9	**16**	**25**	**36**
Zaman Bükülmesi	**Geçen zaman [s]**	1298,46	352,96	474,99	352,10
	Standart sapma (σ) [s]	22,09	0,49	11,92	1,61
Alt-durum Kaydetme	**Geçen zaman [s]**	1277,11	564,74	470,88	351,04
	Standart sapma (σ) [s]	19,27	1,45	16,08	0,88

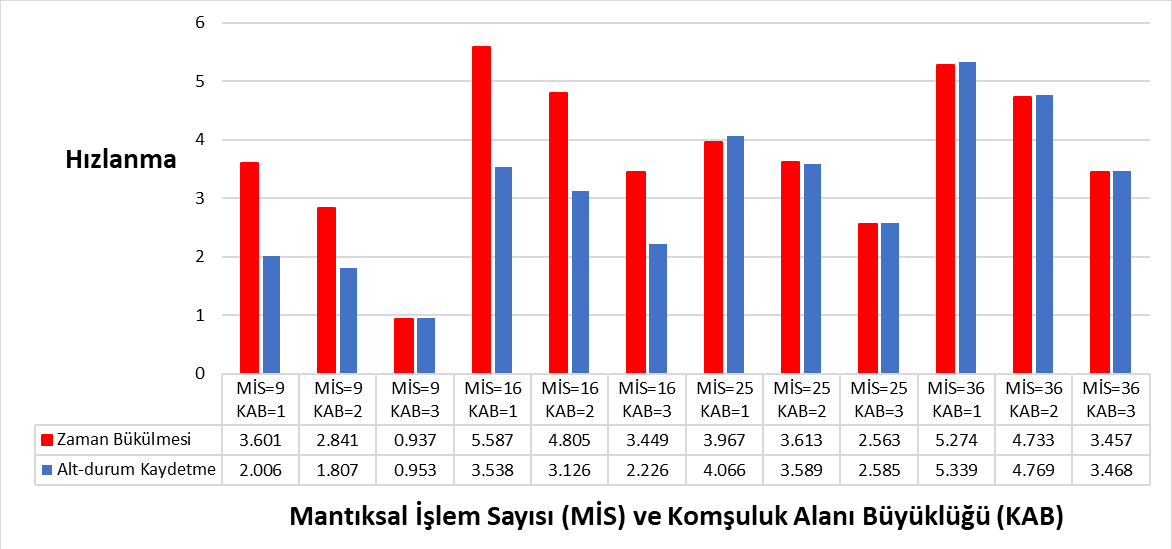


Şekil 6.10. Etmen sayısı 25920 ve komşuluk alanı büyüklüğü 3 hücre iken benzetimin farklı sayılardaki Mİ'ler ile çalıştırılma süresi



	MİS=9 KAB=1	MİS=9 KAB=2	MİS=9 KAB=3	MİS=16 KAB=1	MİS=16 KAB=2	MİS=16 KAB=3	MİS=25 KAB=1	MİS=25 KAB=2	MİS=25 KAB=3	MİS=36 KAB=1	MİS=36 KAB=2	MİS=36 KAB=3
Zaman Bükülmesi	3.310	4.049	3.496	3.757	6.265	5.519	3.575	4.594	4.088	4.493	5.924	5.395
Alt-durum Kaydetme	1.896	2.350	2.065	3.253	4.032	3.583	3.471	4.603	4.098	4.543	5.973	5.396

Şekil 6.11. Etmen sayısı 12960 iken elde edilen hızlanma değerleri



	MİS=9 KAB=1	MİS=9 KAB=2	MİS=9 KAB=3	MİS=16 KAB=1	MİS=16 KAB=2	MİS=16 KAB=3	MİS=25 KAB=1	MİS=25 KAB=2	MİS=25 KAB=3	MİS=36 KAB=1	MİS=36 KAB=2	MİS=36 KAB=3
Zaman Bükülmesi	3.601	2.841	0.937	5.587	4.805	3.449	3.967	3.613	2.563	5.274	4.733	3.457
Alt-durum Kaydetme	2.006	1.807	0.953	3.538	3.126	2.226	4.066	3.589	2.585	5.339	4.769	3.468

Şekil 6.12. Etmen sayısı 25920 iken elde edilen hızlanma değerleri

Elde edilen bu sonuçlar incelendiğinde, özetle şu bulgulara ulaşılmaktadır;

- Benzetim daha az sayıda Mİ ile gerçekleştirildiğinde Zaman Bükülmesi mekanizması daha başarılı sonuçlar vermektedir. Alt-durum kaydetme yönteminin getirmiş olduğu ek yük Zaman Bükülmesi mekanizmasından daha iyi çalışmasını mümkün kılmamıştır.
- Kullanılan Mİ sayısı arttıkça, alt-durum kaydetme mekanizmasının Zaman Bükülmesi mekanizmasını yakaladığı ve küçük farklarla da olsa geçtiği görülmektedir.

- Zaman Bükülmesi mekanizmasının Amdahl yasasının olumsuz etkisiyle karşılaşması [122] noktasında ilginç bir durum söz konusudur. Farklı işlemcilere ait çekirdekler kullanıldığında, Zaman Bükülmesi mekanizmasının koşut çalıştırılmasının başarımı olumsuz etkilediği görülmektedir. Yapılan tüm deneylerde Zaman Bükülmesi mekanizmasının 16 Mİ ile çalıştırıldığı durumda 25 Mİ ile çalıştırıldığı durumdan daha başarılı olduğu gözlenmektedir. Ancak; alt-durum kaydetme mekanizmasının hiçbir deneyde Amdahl yasasının olumsuz etkisiyle karşılaşmadığı görülmektedir. Bu durumda, alt-durum kaydetme mekanizmasının kullanılmasıyla Zaman Bükülmesi mekanizmasının, çoklu düğümün kullanıldığı deneyler için daha ölçeklenebilir sonuçlar verebileceği söylenebilir.
- Komşuluk alanı büyüklüğünün artırılması etmenler arasında daha çok etkileşime olanak sağlanması demektir. Bu nedenle komşuluk alanı büyüklüğü artırıldığında her iki algoritma da benzetimi daha geç sürelerde tamamlayabilmektedir.

Özetle, alt-durum kaydetme mekanizması sayesinde Mİ'lerin mızıkçı iletileri alır almaz geri sarma işlemlerinin gerekliliğini değerlendirebildiği görülmüştür. Böylece gereksiz geri sarma işlemlerinden kaçınılmış ve belli durumlarda alt-durum kaydetme mekanizmasının Zaman Bükülmesi mekanizmasını yakalayabildiği ve küçük farklarla öne geçebildiği görülmüştür. Alt-durum kaydetme mekanizmasının dolaylı avantajlarından birisi de geri sarma sayısının azalmasıyla birlikte Mİ'lerin daha iyimser davranabilmesi ve bunun sonucunda da denetim noktası alma sıklığının azalması olmuştur. Örneğin; Zaman Bükülmesi mekanizmasının kullanıldığı deneylerde denetim noktası aralığı en çok 4 zaman adımı olabilirken, bu değerin alt-durum kaydetme mekanizmasıyla 12'ye çıktığı gözlenmiştir. Alt-durum kaydetme mekanizması ve elde edilen deney sonuçları [127]'de sunulmuştur.

Tüm bu avantajlarına karşın, alt-durum kaydetme mekanizmasının genel amaçlı kullanılabilirliği konusunda birtakım sorunlar bulunmaktadır. Örneğin; tampon alanlardaki etmenlere ait tüm özniteliklerin korunması gerekmemekte, yalnızca geri sarmaların gerekliliğini incelemeye yardımcı olacak özniteliklerin korunması yeterli olmaktadır. Ancak bu özniteliklerin model bağımlı olarak değişiklik göstermesi ve geliştiricilerin durum kaydetme mekanizmasına müdahale etmesini gerektirmesi

yöntemin en belirgin olumsuzluklarından bir tanesidir. Bu nedenle önerilen yöntemin daha da iyileştirilmesi amacıyla Bölüm 7'de ayrıntıları verilen ÖGS mekanizması geliştirilmiştir.

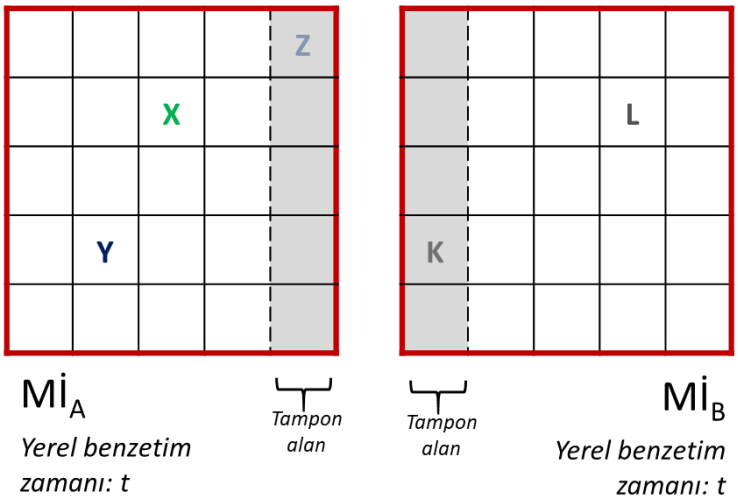
Ayrıca benzetimlerin çalıştırıldığı ortamdaki kaynakların sınırlı olması, yöntemlerin daha yüksek sayıdaki çekirdeklerle/işlemcilerle çalıştırılabilmesini önlemiştir. Bu nedenle daha gelişmiş YBH sistemlerinin kullanılması gerektiği fark edilmiştir.

7. ÖNGÖRÜLMÜŞ GERİ SARMA MEKANİZMASI

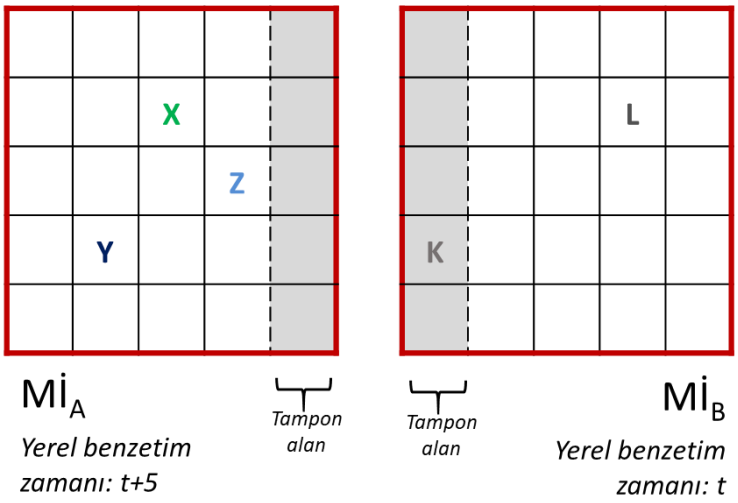
Bölüm 6'da bahsedilen gereksiz geri sarma işlemlerinden kaçınabilecek eklentilerin geliştirilmesi Zaman Bükülmesi mekanizmasının daha da hızlı çalışmasını sağlayacaktır. Bu amaçla Bölüm 6.1'de ayrıntıları verilen durum farkı hesaplama yönteminin çok pahalı olacağı henüz geliştirilmeden görülmüştür. Sonrasında Bölüm 6.2'de önerilen alt-durum kaydetme mekanizması önerilmiş; böylece bir Mİ'nin mızıkçı iletiyi aldığı anda geri sarma işleminin gereksiz olup olmadığını anlayabilmesini sağlayan bir yöntem geliştirilmiştir. Ancak bu yöntemle istenen verim elde edilememiş ve genel amaçlı bir çözüm sunulamamıştır. Bu çözümün daha etkin bir hale getirilmesi için Öngörülmüş Geri Sarma (ÖGS) mekanizması geliştirilmiştir. ÖGS mekanizması da tıpkı alt-durum kaydetme mekanizması gibi, mızıkçı iletilerin alınır alınmaz incelenmesini sağlar. Bu incelemeyi yapabilmek için de Mİ'ler benzetim devam ederken ileriye dönük öngörülerde bulunur. Bu öngörüler, ilerleyen zamanlarda Mİ'nin ne tür mızıkçı iletiler geldiğinde geri sarma işlemi başlatacağını, ne tür mızıkçı iletiler geldiğinde geri sarma işleminden kaçınacağını belirlemeye yöneliktir.

Şekil 7.1 ve Şekil 7.2'de Zaman Bükülmesi mekanizması kullanıldığında ortaya çıkabilecek gereksiz bir geri sarma örneği verilmiştir. Bu örnekte etmenlerin yaşadığı 2 boyutlu ızgara uzayı $Mİ_A$ ve $Mİ_B$ tarafından yönetilmektedir. Her bir Mİ sahip olduğu etmenleri yönetmekte ve tampon bölgede bulunan etmenleri komşusu olan Mİ'ye bildirmektedir. Eğer sakıngan yönetim mekanizması kullanılsaydı (veya benzetim koşut çalıştırılmasaydı), benzetim durumu Şekil 7.1'de görüldüğü gibi olacaktı. Ancak Zaman Bükülmesi mekanizması kullanıldığında Mİ'lerin zaman uyumlamasının gevşetilmesi sonucu Şekil 7.2'dekine benzer bir durum ortaya çıkabilecekti. Şekil 7.2'de $Mİ_A$'nın yerel zamanı $t + 5$ iken, $Mİ_B$'nin yerel zamanının t olduğu görülmektedir. $Mİ_B$ tampon alanında bulunan K etmenini $Mİ_A$'ya bildirdiğinde, $Mİ_A$ daha ileri bir zamanda olduğu için gelen bu iletiyi mızıkçı ileti olarak değerlendirir. Zaman Bükülmesi mekanizması bu durumda doğrudan bir geri sarma işlemi başlatır. Bu örnek için komşuluk alanı büyüklüğünün 1 hücre olduğu varsayılsın. Eğer K etmeni doğru zamanda (her iki Mİ'nin yerel zamanı t iken) iletilseydi, K ve Z etmenleri aralarındaki mesafe dolayısıyla etkileşime

giremeyecekti. Ancak Zaman Bükülmesi mekanizmasında Mİ$_A$'nın t anında bu durumu bilmesi mümkün değildir. Çünkü Mİ$_A$ $t+5$ anında, t zamanındaki geçmiş benzetim durumuna sahip değildir. ÖGS mekanizması tıpkı alt-durum kaydetme mekanizmasında olduğu gibi bu tür mızıkçı iletilerin gerçekten bir geri sarmaya neden olup olmayacağını inceleyebilmek üzere geliştirilmiştir. Dolayısıyla ÖGS mekanizmasında bir mızıkçı ileti alan Mİ'nin izleyeceği akış Şekil 6.2'deki gösterim ile aynı şekilde gerçekleşecektir. ÖGS mekanizmasının kaçınmaya çalıştığı geri sarma işlemleri Bölüm 5.3.1'de bahsedilen "etmenlerin komşu Mİ'lere görünürlüğü" türünden olaylardır. Diğer türdeki olaylar için geri sarma işlemlerinden kaçınılmaz.



Şekil 7.1. Zaman uyumlama yapılmış olsaydı Mİ$_A$ ve Mİ$_B$'nin sahip olacakları durum



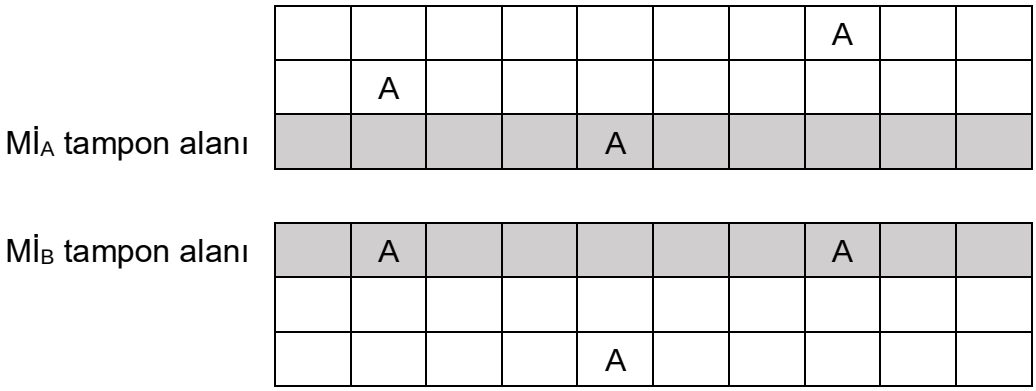
Şekil 7.2. Zaman uyumlama yapılmamış olduğu için Mİ$_A$ ve Mİ$_B$'nin sahip oldukları durum

Yukarıda da bahsedildiği üzere, ÖGS mekanizması gereksiz geri sarma işlemlerini tespit edip bunlardan kaçınmayı sağlamak üzere Zaman Bükülmesi mekanizmasına bir eklenti olarak geliştirilmiştir. Kaçınılan geri sarma işlemleri benzetim sonuçlarına etki etmeyen gereksiz geri sarma işlemleridir. Dolayısıyla benzetimin doğruluğu ve tekrarlanabilirliği bu kaçınma işlemleri sonucunda zarar görmez. Zaman Bükülmesi mekanizmasında en çok zaman alan adımlardan bir tanesi olan geri sarma işlemlerinin gereksiz olanlarından kaçınılması kayda değer bir şekilde benzetimlerin daha hızlı bir şekilde tamamlanmasını sağlayacaktır. Bu amaçla uzamsal-koşut benzetimlere yönelik olarak geliştirilen ÖGS mekanizması, mızıkçı iletileri değerlendirerek geri sarma işleminin gerekliliğine karar veren genel amaçlı bir yöntem sunmaktadır.

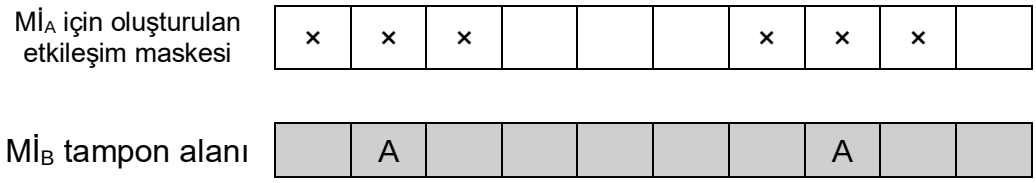
ÖGS mekanizmasıyla geliştirilen RHPC_PR'ye ait sınıf şemasının özet bir hali Ek - 4'te verilmiştir.

7.1. Öngörülmüş Geri Sarma Mekanizmasının Temel İlkeleri

ÖGS mekanizmasının temelinde etkileşim maskeleri ve mızıkçı ileti izleri adı verilen veri yapıları yer almaktadır. Etkileşim maskeleri, Mİ'nin o anki durumu göz önünde bulundurularak gelecekte ortaya çıkabilecek etkileşimleri kestirmek üzere ilgili Mİ tarafından oluşturulurlar. Bir başka deyişle etkileşim maskeleri "İleri bir zamanda, nasıl bir mızıkçı ileti alınırsa geri sarma işlemi yapılmak zorunda kalınır?" sorusunun cevabını içerecek şekilde hazırlanır. Böylece ileri bir zamanda alınacak mızıkçı iletilerin değerlendirilebilmesi için erken bir zamanda önlem alınmış olur. Bir Mİ'nin tampon alandaki benzetim durumu her değiştiğinde bir etkileşim maskesi oluşturulur ve bir sonraki denetim noktasına kadar silinmez. Her bir Mİ'nin $N \times N$'lik bir alt-uzayı yönettiği ve komşuluk alanı büyüklüğünün b olduğu uzamsal-koşut bir benzetimde, etkileşim maskeleri $b \times N$ boyutlarında hazırlanır. Her bir etmen için, o etmenin komşuluk alanı maske üzerinde işaretlenir. Şekil 7.3'te iki Mİ tarafından yönetilen, her bir Mİ'nin yönettiği alt-uzay büyüklüğünün 3×10 ve komşuluk alanı büyüklüğünün 1 olduğu bir benzetimin t anındaki görünümü örneklenmiştir. t anında her iki Mİ de tampon alanında etmenler bulundurduğu için birer etkileşim maskesi oluşturacaktır. $Mİ_B$'nin $Mİ_A$'dan t zaman damgasıyla gelebilecek mızıkçı iletilere karşı önlem almak için oluşturacağı etkileşim maskesi Şekil 7.4'te verilmiştir.

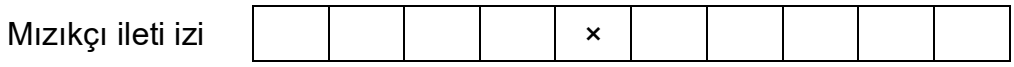


Şekil 7.3. İki Mİ tarafından yönetilen bir benzetimin t anındaki görünümü



Şekil 7.4. MİB'nin MİA'dan gelebilecek t zaman damgalı mızıkçı iletiler için oluşturduğu etkileşim maskesi

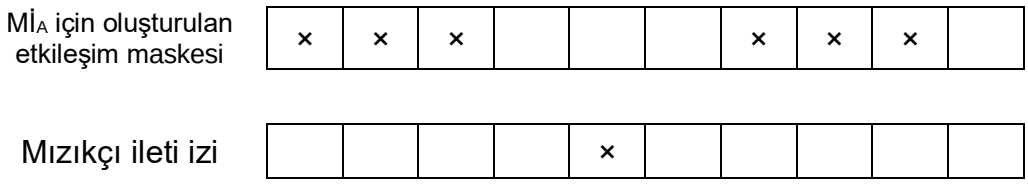
Mızıkçı ileti izleri ise bir mızıkçı iletiyi alan Mİ tarafından oluşturulur ve etkileşim maskesi ile bir maskeleme işlemi yapmak için kullanılır. Bu maskeleme işlemi sonucunda ortaya çıkan duruma göre mızıkçı iletinin geçmiş bir zamanda etkileşime neden olup olmayacağı belirlenerek geri sarma işleminin gerekliliğine karar verilir. Örneğin; Şekil 7.3'te verilen durumda t anında MİA tampon alanında bulunan iletileri MİB'ye gönderir. MİB bu iletiyi t anından daha geç bir zamanda aldığında bu iletiyi mızıkçı ileti olarak görür ve Şekil 7.5'te görülen mızıkçı ileti izini oluşturur.



Şekil 7.5. MİB'nin MİA'dan gelen t zaman damgalı mızıkçı ileti için oluşturduğu mızıkçı ileti izi

Yukarıda verilen örnekte, mızıkçı iletinin etkileşime neden olup olmayacağı ise maskeleme işleminin sonrasında ortaya çıkar. Maskeleme işlemi, etkileşim maskesinin mızıkçı ileti iziyle eşleştirilmesiyle gerçekleştirilir. Şekil 7.6'da gösterilen maskeleme işlemi sonucunda etkileşim maskesindeki hiçbir hücrenin, mızıkçı ileti izinde bir karşılığı olmadığı görülmektedir. Bu durumda mızıkçı ileti zamanında

alınsaydı bile, etmenler bulunduğu konum dolayısıyla etkileşime girmeyeceklerdi. Sonuç olarak bu mızıkçı ileti alındığında geri sarma işlemi yapmak gerekli olmayacaktı.

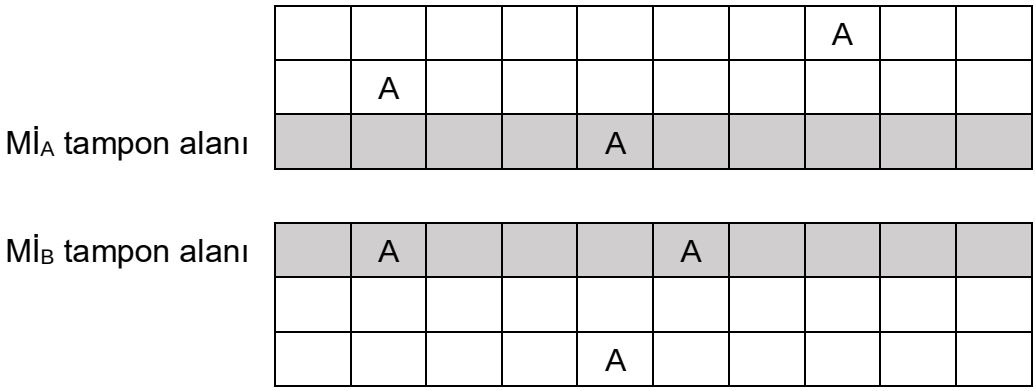


Şekil 7.6. Maskeleme işlemi sonucunda geri sarmanın gereksiz olduğuna karar verilmesi

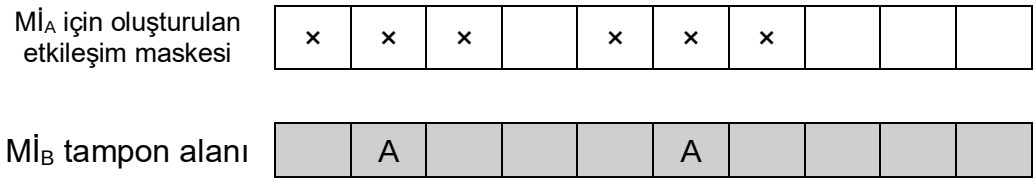
Geri sarma işleminin kaçınılmaz olduğu bir örnek Şekil 7.7'de görülebilir. Bu örnekte Mİ$_A$'nın tampon alanındaki iletilerin Mİ$_B$'ye olması gerekenden daha geç bir zamanda bildirildiği varsayılsın. Mİ$_B$ t anında benzetimi işletirken Şekil 7.8'deki gibi bir etkileşim maskesi oluşturur. Daha sonraki bir zamanda t zaman damgalı bir mızıkçı ileti aldığında ise önce Şekil 7.9'daki gibi bir mızıkçı ileti izi oluşturur ve Şekil 7.10'daki maskeleme işlemini gerçekleştirir. Maskeleme işlemi sonucunda etkileşim maskesindeki en az bir hücrenin, mızıkçı ileti izinde bir karşılığı olduğu görülmektedir. İlgili hücrede bulunan etmen(ler), mızıkçı ileti zamanında alınsaydı bu iletinin içereceği etmenlerle etkileşime girebilecekti. Bir başka deyişle, Mİ$_B$'nin tampon alanında bulunan bazı etmenlerin geçmiş zamanda gerçekleştirdiği davranış daha farklı olabilirdi. Dolayısıyla mızıkçı iletinin zamanında alınamaması benzetim durumunun olması gerekenden farklı bir noktaya gelmesine neden olmuştur ve bir geri sarma işlemi yapılarak bu hatanın telafi edilmesi gerekmektedir.

7.2. Öngörülmüş Geri Sarma Mekanizmasının Etkileşim Modeli Kullanılarak İyileştirilmesi

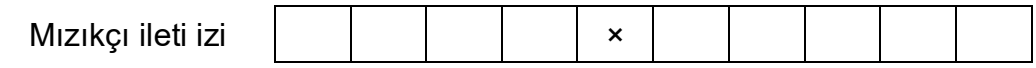
Bu bölüme kadar anlatılan kısımda etmenler arasındaki etkileşimin gerçekleşmesi sadece aralarındaki mesafeye bağlı olarak değerlendirilmiştir. Halbuki bir etmen tabanlı modelde etmenlerin birbirine yakın olması etkileşimde bulunacakları anlamına gelmez. Dolayısıyla benzetimi yapılacak modelin etkileşim modeli kullanılarak ÖGS mekanizmasının iyileştirilmesi mümkündür. Bu iyileştirmeyle birlikte geri sarma işlemlerinin gerekliliği etmenler arasında gerçekten etkileşim olup olmayacağına göre değerlendirilebilir.



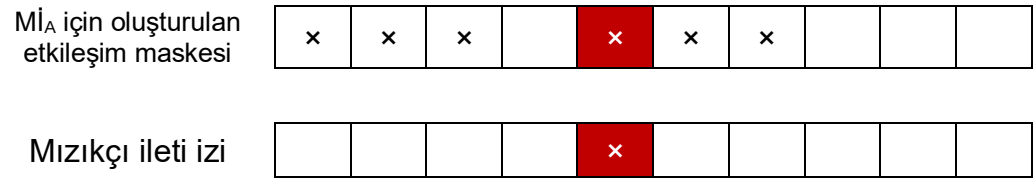
Şekil 7.7. İki Mİ tarafından yönetilen bir benzetimin t anındaki görünümü



Şekil 7.8. Mİ$_B$'nin Mİ$_A$'dan gelebilecek t zaman damgalı mızıkçı iletiler için oluşturduğu bir etkileşim maskesi



Şekil 7.9. Mİ$_B$'nin Mİ$_A$'dan gelen t zaman damgalı mızıkçı ileti için oluşturduğu mızıkçı ileti izi



Şekil 7.10. Maskeleme işlemi sonucunda geri sarmanın kaçınılmaz olduğuna karar verilmesi

Etmenler arası etkileşime ek olarak, etmenler ve ortam arasında da bir etkileşim olabilir. Örneğin bir yiyecek arama modelinde etmenler yiyeceklerin bulunduğu ortamı keşfetmek üzere tasarlanmış olabilir. Böyle bir örnekte yiyecekler de birer etmen olarak tanımlanıp ÖGS mekanizmasının önerdiği etkileşim modeline dahil edilebilir. Bunun yanı sıra ortam, benzetim ortamına yayılmış özellikleri de temsil edebilir. Örneğin; benzetim ortamının belirli bir bölgesine ait bir sıcaklık verisi söz

konusu olabilir. Bu tür durumlarda da ortam, uzayın ilgili bölgesinde yaşayan özel bir etmen (örneğin 2 boyutlu bir uzayın her hücresinde yer alan bir etmen) olarak tanımlanabilir. Sonuç olarak etmenler ve ortam arasındaki ilişkinin, ortamın etmen olarak tanımlanmasıyla ÖGS mekanizması kapsamında ele alınması mümkündür.

ÖGS mekanizmasının etkileşim modeli kullanılarak iyileştirilmesi durumunda maskeleme, etkileşim maskesi ve mızıkçı ileti izi oluşturma yöntemlerinde bazı değişiklikler yapılması gerekmektedir. Bu tez çalışması kapsamında gerçekleştirilen yöntem asal sayılardan yararlanmaktadır. Asal sayıların, kendisi ve 1 dışında başka iki sayının çarpılmasıyla elde edilemeyecek olmasından dolayı etmenler arasındaki etkileşimin varlığının tespit edilmesinde başarılı bir şekilde görev alabileceği değerlendirilmiştir. Etkileşimin varlığının başka yöntemlerle de tespit edilmesi mümkün olabilir. Ancak bu çalışmada bir gerçekleştirim kararı olarak asal sayıların kullanımının yeterli olacağı ve etkin bir çözüm sunacağı değerlendirilmiştir. Buna göre etkileşim modelini tanımlamada kullanılacak her bir etmen türü için özgün (İng. *unique*) bir asal sayı belirlenir. Buna göre bir etkileşim maskesi şu kurallara göre oluşturulur:

1. $b \times N$ boyutlarında bir etkileşim maskesi yaratılır.
2. Tampon alandaki her bir etmen için;
    a. Etkileşim maskesindeki komşuluk alanı belirlenir.
    b. Etkileşim maskesindeki komşuluk alanında bir değer mevcutsa, bu değer o anki etmenin etkileşime girebileceği etmen türlerini tanımlayan asal sayı ile çarpılır.
    c. Etkileşim maskesindeki komşuluk alanında bir değer mevcut değilse, o alana o anki etmenin etkileşime girebileceği etmen türlerini tanımlayan asal sayıların çarpımı konulur.

Önerilen yöntem, daha iyi anlaşılabilmesi için Chemical Equilibrium [128] adı verilen bir moleküler benzetim örneği üzerinden açıklanmıştır. Chemical Equilibrium modelinde CO, CO_2, NO ve NO_2 olmak üzere dört molekülün zaman içerisinde birbirleriyle etkileşimi incelenmektedir. Bu modeli basit şekilde açıklamak gerekirse;

- CO ve NO_2 etmenleri birbirlerine yeterince yaklaştığında tepkimeye girerek CO_2 ve NO moleküllerini oluşturmaktadır.

- NO ve CO_2 etmenleri birbirlerine yeterince yaklaştığında tepkimeye girerek NO_2 ve CO moleküllerini oluşturmaktadır.
- Örneği basit tutmak adına tepkime için gerekli aktivasyon enerjisi ve ortaya çıkacak enerjinin hesaplanması gibi sayısal işlemler göz ardı edilmiştir.

Modelin bu özellikleri göz önünde bulundurulduğunda şu şekilde bir etkileşim modelinin bulunduğu görülmektedir:

- CO etmenleri NO_2 etmenleriyle etkileşime girebilirler.
- CO_2 etmenleri NO etmenleriyle etkileşime girebilirler.
- NO etmenleri CO_2 etmenleriyle etkileşime girebilirler.
- NO_2 etmenleri CO etmenleriyle etkileşime girebilirler.

Bu etkileşim modeline göre, her bir molekül yalnızca belli bir tür molekülle etkileşime girmektedir. Örneğin CO molekülleri CO_2 ve NO molekülleriyle etkileşime girmemektedir. Dolayısıyla CO moleküllerinin davranışı etrafındaki CO_2 ve NO moleküllerinin varlığından etkilenmeyecektir. ÖGS mekanizmasına göre her bir etmen türüne birbirinden farklı bir asal sayı rastgele bir şekilde atanır. Örneğin; bu model için en küçük dört asal sayının CO etmenleri için 2, CO_2 etmenleri için 3, NO etmenleri için 5 ve NO_2 etmenleri için ise 7 olacak şekilde atandığı varsayılsın. Bu atama ve etkileşim modeli göz önünde bulundurulduğunda CO etmenleri 7, CO_2 etmenleri 5, NO etmenleri 3 ve NO_2 etmenleri ise 2 ile temsil edilen etmen türleriyle etkileşim halindedir.

Şekil 7.11'de komşuluk alanı büyüklüğünün 1 olduğu bir benzetimde t anındaki iki komşu Mİ'nin tampon alanlarının görünümü örneklenmiştir. t anında her iki Mİ de tampon alanında etmenler bulundurduğu için birer etkileşim maskesi oluşturacaktır. $Mİ_B$'nin $Mİ_A$'dan t zaman damgasıyla gelebilecek mızıkçı iletilere karşı önlem almak için oluşturacağı etkileşim maskesi Şekil 7.12'de verilmiştir. Eğer $Mİ_A$'dan t zaman damgasıyla gönderilen ileti $Mİ_B$ tarafından t anından daha geç bir zamanda alınırsa $Mİ_B$, Şekil 7.13'teki gibi bir mızıkçı ileti izi oluşturur. Ardından $Mİ_B$ Şekil 7.14'te görüldüğü üzere etkileşim maskesini mızıkçı ileti izine böler. Bu bölme işlemi sonucunda etkileşim maskesindeki bazı hücrelerin mızıkçı ileti izindeki bazı hücrelere tam bölünebildiği görülmektedir. Bu tam bölünebilme durumu, Şekil

7.11'de görülen Mİ$_A$'daki CO etmeni ile Mİ$_B$'deki NO_2 etmeninin ve Mİ$_A$'daki NO etmeni ile Mİ$_B$'deki CO_2 etmeninin birbirine yeterince yakın olması ve tepkimeye girebilmesi olasılığından kaynaklanmaktadır. Dolayısıyla bu örnekte geri sarma işleminin kaçınılmaz olduğu anlaşılmaktadır.

Mİ$_A$ tampon alanı	NO			CO					NO	
Mİ$_B$ tampon alanı		CO			NO_2				CO_2	

Şekil 7.11. Chemical Equilibrium modelini yöneten iki Mİ'nin t anındaki görünümü

Mİ$_A$ için oluşturulan etkileşim maskesi	7	7	7	2	2	2		5	5	5
Mİ$_B$ tampon alanı		CO			NO_2				CO_2	

Şekil 7.12. Mİ$_B$'nin Mİ$_A$'dan gelebilecek t zaman damgalı mızıkçı iletiler için oluşturduğu etkileşim maskesi

Mızıkçı ileti izi	5			2					5	

Şekil 7.13. Mİ$_B$'nin Mİ$_A$'dan gelen t zaman damgalı mızıkçı ileti için oluşturduğu iz

Mİ$_A$ için oluşturulan etkileşim maskesi	7	7	7	2	2	2		5	5	5
				÷						
Mızıkçı ileti izi	5			2					5	

Şekil 7.14. Maskeleme işlemi sonucunda geri sarmanın kaçınılmaz olduğuna karar verilmesi

Eğer, Mİ$_A$'dan gelen mızıkçı ileti ve bu mızıkçı ileti için Mİ$_B$'nin oluşturduğu mızıkçı ileti izi Şekil 7.15'deki gibi olsaydı maskeleme işlemi de Şekil 7.16'da gösterildiği gibi gerçekleşecekti. Bu maskeleme işlemi sonucunda etkileşim maskesi ve mızıkçı ileti izinin birbirine karşılık gelen herhangi bir hücresinde tam bölünebilme işlemi gerçekleşemeyeceği için geri sarmanın gereksiz olduğu anlaşılacaktı.

$Mİ_A$'dan gelen mızıkçı ileti	NO			CO_2					CO	
Mızıkçı ileti izi	5			3					2	

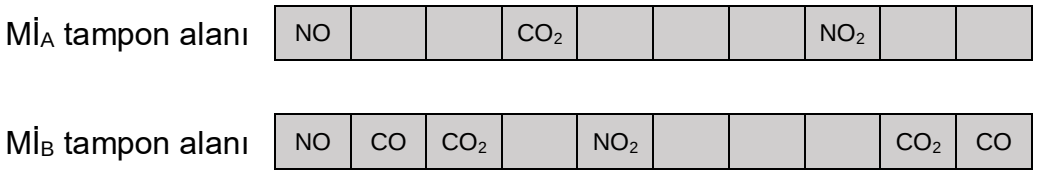
Şekil 7.15. $Mİ_B$'nin $Mİ_A$'dan aldığı t zaman damgalı mızıkçı ileti ve bu ileti için oluşturduğu mızıkçı ileti izi

$Mİ_A$ için oluşturulan etkileşim maskesi	7	7	7	2	2	2		5	5	5
						÷				
Mızıkçı ileti izi	5			3					2	

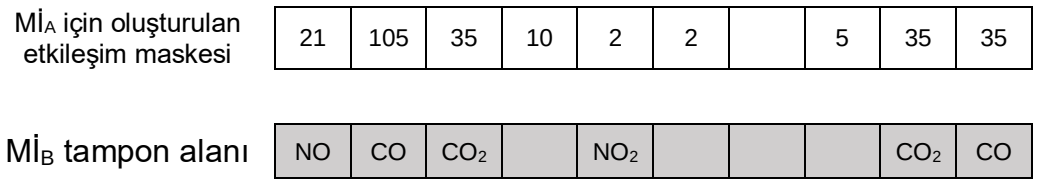
Şekil 7.16. Maskeleme işlemi sonucunda geri sarmanın gereksiz olduğuna karar verilmesi

Chemical Equilibrium modelinde etkileşim modelinin kullanılarak maskeleme işleminin nasıl yapıldığına yönelik daha karmaşık bir örnek de Şekil 7.17 – Şekil 7.20'de verilmiştir. Bu örnekte yan yana bulunan etmenler için etkileşim maskesinin nasıl oluşturulduğu örneklenmiştir. Maskeleme işlemi sonucunda geri sarma işleminin gereksiz olduğu anlaşılacaktır. Geri sarma işleminin kaçınılmaz olduğu bir başka örnek ise Şekil 7.21 – Şekil 7.24'te verilmiştir.

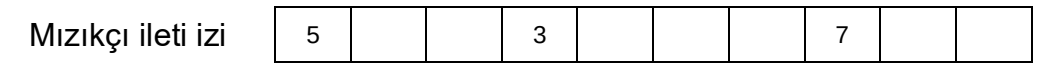
Chemical Equilibrium modeline benzer şekilde bir çok uzamsal-koşut etmen tabanlı modelin de ÖGS mekanizmasını kullanması mümkündür. Böylece benzetimlerin çalıştırılma sürelerinde kısalma beklenmektedir. Hastalık yayılım modelleri [109], orman yangını modelleri [129], sosyolojide kullanılan ırkçılık modelleri [130] bu örneklerden sadece bir kaçıdır. ÖGS mekanizmasının kullanılabileceği modellere ait ayrıntılı bir değerlendirme Bölüm 7.6'da verilmiştir.



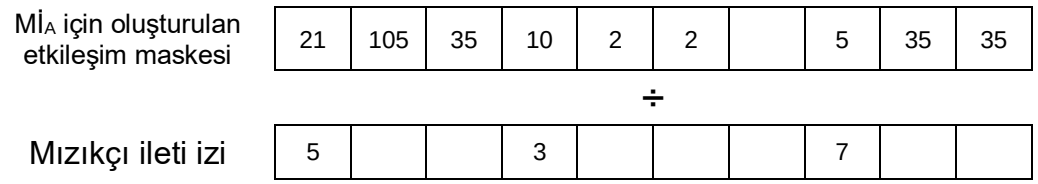
Şekil 7.17. Chemical Equilibrium modelini yöneten iki Mİ'nin t anındaki görünümü



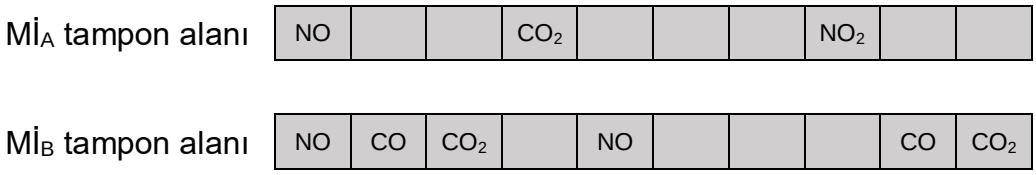
Şekil 7.18. Mİ$_B$'nin Mİ$_A$'dan gelebilecek t zaman damgalı mızıkçı iletiler için oluşturduğu etkileşim maskesi



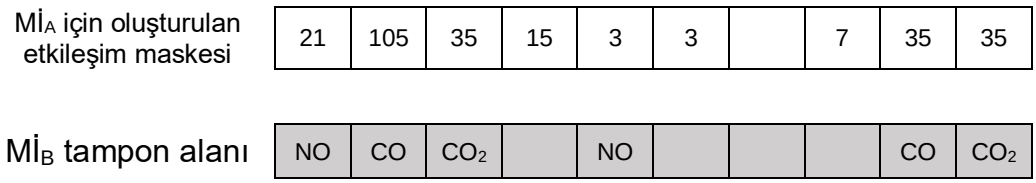
Şekil 7.19. Mİ$_B$'nin Mİ$_A$'dan aldığı t zaman damgalı mızıkçı ileti için oluşturduğu mızıkçı ileti izi



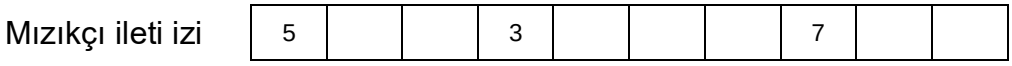
Şekil 7.20. Maskeleme işlemi sonucunda geri sarmanın gereksiz olduğuna karar verilmesi



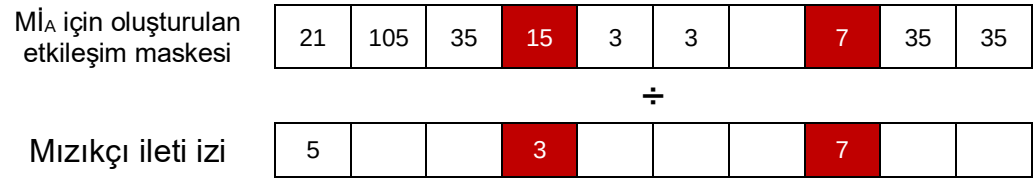
Şekil 7.21. Chemical Equilibrium modelini yöneten iki Mİ'nin t anındaki görünümü



Şekil 7.22. Mİ$_B$'nin Mİ$_A$'dan gelebilecek t zaman damgalı mızıkçı iletiler için oluşturduğu etkileşim maskesi



Şekil 7.23. Mİ$_B$'nin Mİ$_A$'dan aldığı t zaman damgalı mızıkçı ileti için oluşturduğu mızıkçı ileti izi



Şekil 7.24. Maskeleme işlemi sonucunda geri sarmanın kaçınılmaz olduğuna karar verilmesi

7.3. Durum Çalışması

ÖGS mekanizmasının Zaman Bükülmesi mekanizmasıyla karşılaştırılması için Civil Violence modeli [120] kullanılmıştır. Bölüm 6.2.1'de Civil Violence modelinin etkileşim modeli açıklanmıştır. Bu etkileşim modeli ÖGS mekanizması için kullanıldığında her bir etmen türü için özgün birer asal sayı atanır. Bu asal sayılar polis etmenleri (C) için 2, sakin etmenler (Q) için 3, eylemci etmenler (A) için 5 ve mahkum etmenler (J) için 7 olarak belirlenmiştir. Bu belirleme tamamen rastgele olarak yapılmış olup, etmen türlerine farklı asal sayılar da atanabilirdi. Yapılan bu atamaya göre etkileşim modeli şu şekilde özetlenebilir:

- C etmenleri etrafında bulunan A (5) etmenleri ile etkileşime girebilirler.

- Q etmenleri etrafında bulunan C (2) ve A (5) etmenleri ile etkileşime girebilirler.
- A etmenleri etrafında bulunan C (2) etmenleri ile etkileşime girebilirler.
- J etmenleri herhangi bir etmenle etkileşime giremezler.

Bu etkileşim modeli göz önünde bulundurularak yapılan örnek bir maskeleme işlemi Şekil 7.25 – Şekil 7.28'de verilmiştir. Şekil 7.25 iki Mİ tarafından yürütülen, her bir Mİ'nin 3×10'luk bir alanı yönettiği ve komşuluk alanı büyüklüğünün 1 olduğu bir benzetimin t anındaki görünümünü örneklemektedir. Şekil 7.26 $Mİ_B$'nin t anında $Mİ_A$'dan ileride gelebilecek t zaman damgalı bir mızıkçı iletiye karşı önlem almak için oluşturduğu etkileşim maskesini göstermektedir. $Mİ_B$, $Mİ_A$'dan mızıkçı bir ileti aldığında ise Şekil 7.27'de gösterilen mızıkçı ileti izini oluşturmaktadır. Daha sonra Şekil 7.28'de gösterilen maskeleme işlemini yapmak üzere etkileşim maskesi mızıkçı ileti izine bölünür. Bu bölme işleminde etkileşim maskesindeki herhangi bir hücre, mızıkçı ileti izinde karşılık gelen hücreye tam olarak bölünememektedir. Bu da geri sarma işleminin gereksiz olduğunu göstermektedir.

Şekil 7.29 – Şekil 7.32'de kaçınılamayan bir geri sarma işleminin örneği verilmiştir. Bu örnekte Mİ'lerin t anındaki görünümü Şekil 7.29'da verilmiştir. Şekil 7.30'da $Mİ_B$'nin $Mİ_A$'dan gelebilecek t zaman damgalı mızıkçı iletiler için oluşturduğu etkileşim maskesi verilmiştir. $Mİ_B$, $Mİ_A$'dan bir mızıkçı ileti aldığında oluşturduğu mızıkçı ileti izi Şekil 7.31'de gösterilmiştir. Şekil 7.32'de ise mızıkçı iletinin gerçekten bir geri sarmaya neden olup olmayacağını anlamak için yapılan maskeleme işlemi gösterilmiştir. Bu maskeleme işlemi sonucunda etkileşim maskesindeki en az bir hücredeki değerin, mızıkçı ileti izinde karşılık geldiği hücredeki değere tam olarak bölünebildiği görülmektedir. Bu etkileşim aslında Şekil 7.29'dan da görülebilecek olan; $Mİ_B$'nin yönettiği Q etmeninin, $Mİ_A$'dan gelecek olan A etmeniyle etkileşime girebilecek olmasından kaynaklanmaktadır.

			A				C			
					J				A	
Mİ$_A$ tampon alanı		Q	J				C			Q

Mİ$_B$ tampon alanı			Q				C		A	
		A		C				J		
					Q				A	

Şekil 7.25. İki Mİ tarafından yönetilen bir benzetimin t anındaki görünümü

Mİ$_A$ için oluşturulan etkileşim maskesi		10	10	10		5	5	10	2	2
Mİ$_B$ tampon alanı			Q				C		A	

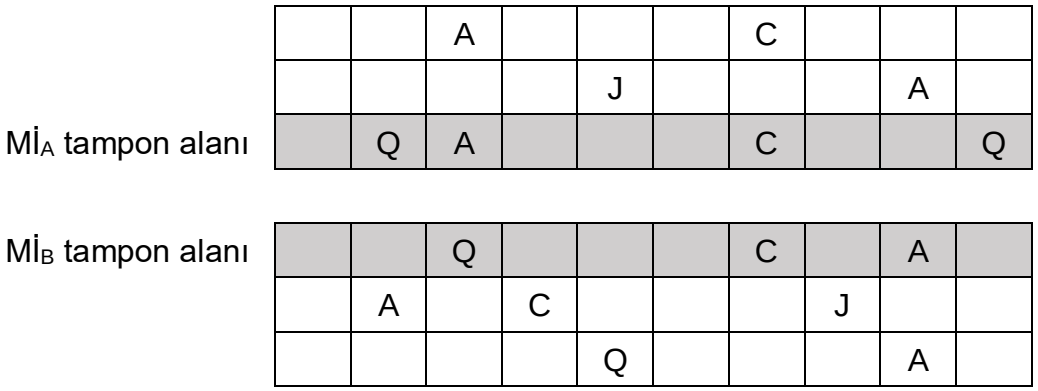
Şekil 7.26. Mİ$_B$'nin Mİ$_A$'dan gelebilecek t zaman damgalı mızıkçı iletiler için oluşturduğu etkileşim maskesi

Mİ$_A$'dan gelen mızıkçı ileti		Q	J				C			Q
Mızıkçı ileti izi		3	7				2			3

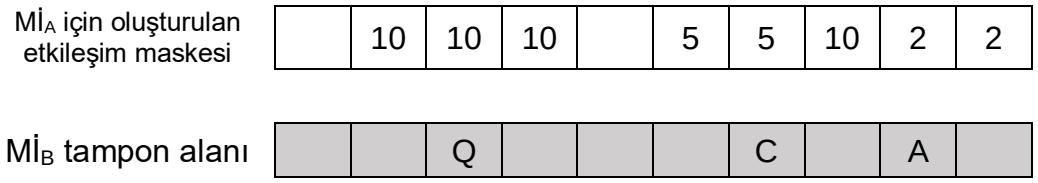
Şekil 7.27. Mİ$_B$'nin Mİ$_A$'dan gelen t zaman damgalı mızıkçı ileti için oluşturduğu mızıkçı ileti izi

Mİ$_A$ için oluşturulan etkileşim maskesi		10	10	10		5	5	10	2	2
						÷				
Mızıkçı ileti izi		3	7				2			3

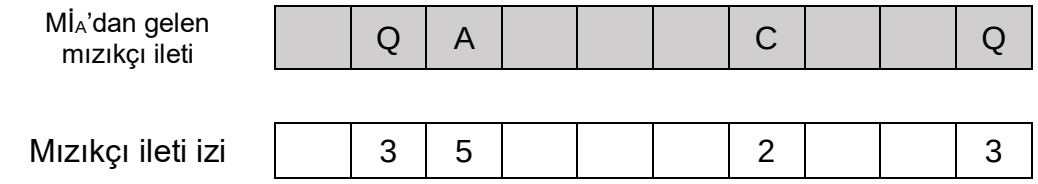
Şekil 7.28. Maskeleme işlemi sonucunda geri sarmanın gereksiz olduğuna karar verilmesi



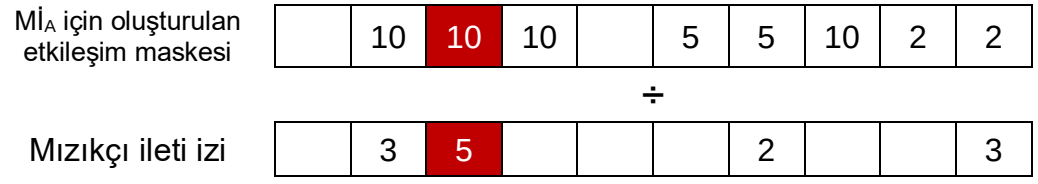
Şekil 7.29. İki Mİ tarafından yönetilen bir benzetimin t anındaki görünümü



Şekil 7.30. Mİ$_B$'nin Mİ$_A$'dan gelebilecek t zaman damgalı mızıkçı iletiler için oluşturduğu etkileşim maskesi



Şekil 7.31. Mİ$_B$'nin Mİ$_A$'dan gelen t zaman damgalı mızıkçı ileti için oluşturduğu mızıkçı ileti izi



Şekil 7.32. Maskeleme işlemi sonucunda geri sarmanın kaçınılmaz olduğuna karar verilmesi

7.4. Deney Ortamı

ÖGS mekanizmasının uzamsal-koşut benzetimlere getirdiği iyileştirmeyi ölçebilmek için Zaman Bükülmesi mekanizmasıyla karşılaştırma yapılmıştır. Deneylerde benzetim yazılımı olarak ÖGS mekanizmasıyla genişletilmiş Repast HPC yazılımı

(RHPC_PR) kullanılmıştır. Deneylerin tamamında TÜBİTAK tarafından sunulan TRUBA YBH altyapısı kullanılmıştır. TRUBA YBH altyapısındaki her bir düğüm 2 adet Intel Xeon Scalable 6148 işlemci içermektedir. Bu işlemcilerin her biri ise 20 çekirdek içermektedir. Düğümler arasındaki ağ altyapısında 100 Gps EDR Infiniband teknolojisi kullanılmaktadır. TRUBA YBH altyapısı farklı MPI gerçekleştirimlerini kullanmayı desteklemektedir. Bu çalışmadaki deneylerde ise Open MPI 1.8.8 gerçekleştirimi kullanılmıştır. TRUBA YBH sisteminde CentOS 7 işletim sistemi kullanılmaktadır.

Deneyler 16, 36, 64, 100 ve 144 özdeş çekirdek ve bu çekirdeklerin yer alabileceği en az sayıda düğüm kullanılarak gerçekleştirilmiştir. Uzamsal ortamın çekirdeklere kolay eşlenebilmesi için çekirdek sayıları bu şekilde seçilmiştir. Yani 16 ve 36 çekirdek kullanıldığında bu çekirdeklerin 1 düğüm üzerinde yer alması; 64 çekirdek kullanıldığında bu çekirdeklerin 2 düğüm üzerinde yer alması; 100 çekirdek kullanıldığında bu çekirdeklerin 3 düğüm üzerinde yer alması; 144 çekirdek kullanıldığında bu çekirdeklerin 4 düğüm üzerinde yer alması sağlanmıştır. 144'ten fazla çekirdek kullanılmak istendiğinde TRUBA sunucularındaki iş kuyruğunda sıranın çok geç gelmesinden veya hiç gelememesinden dolayı daha yüksek sayıda çekirdek kullanılamamıştır. Ancak Bölüm 7.5'te de görülebileceği üzere deneylerin sağlıklı bir şekilde yorumlanabilmesi için 144 çekirdek yeterli olmuştur. TRUBA YBH altyapısındaki çekirdekler çoklu-iş parçacıklı olarak kullanılabilse de bu çalışma kapsamında yapılan deneylerde karşılaştırması yapılan her iki algoritma için de bu teknolojiden faydalanılmamıştır. Her iki algoritma da tamamen aynı koşullarda çalıştırılmış olup bu koşullar şu şekilde verilebilir:

- Uzay boyutu 3600×3600 olarak seçilmiştir.
- Komşuluk alanı büyüklüğü 1 olarak seçilmiştir.
- Başlangıçtaki etmen sayısı 12960 ve 129600 olarak farklı iki yapılandırma belirlenmiştir. Bu çalışmanın kalan kısmında 12960 etmenle yapılan deneyler düşük etmen yoğunluklu, 129600 etmenle yapılan deneyler ise yüksek etmen yoğunluklu deneyler olarak adlandırılacaktır.
- Tüm benzetimler ilk olarak yeterince uzun ve farklı benzetim adımları boyunca çalıştırılmıştır. Elde edilen sonuçlar benzer eğilimleri takip ettiği için yalnızca 1000 benzetim adımı boyunca elde edilen sonuçlara yer verilmiştir.

Uzamsal koşut etmen tabanlı modellerde her bir Mİ'ye düşen işlem yoğunluğu o Mİ'ye düşen etmen sayısıyla ilgilidir. Bu modellerde problem büyüklüğünü artırmak üç şekilde düşünülebilir:

- Etmen sayısı ve uzay büyüklüğünü aynı oranda artırmak,
- Etmen sayısını artırırken uzay boyutunu sabit tutmak,
- Etmen sayısını sabit tutarken uzay boyutunu artırmak.

İyimser zaman yönetim mekanizmasının kullanıldığı modellerde mızıkçı iletilerin ortaya çıkma olasılığı, Mİ'lerin tampon alanlarında etmen bulunma olasılığına bağlı olarak değişecektir. Çünkü mızıkçı iletilerin oluşması, tampon alandaki etmen bilgilerinin ilgili Mİ'lere zamanında ulaştırılamamasından kaynaklanmaktadır. Mİ'lerin tampon alanlarında etmen bulundurma olasılığı ilk seçenek tercih edildiğinde farklı problem büyüklükleri de kullanılsa benzerlik gösterecektir. İkinci ve üçüncü seçenekler ise aslında aynı amaca hizmet etmekte olup, problem büyüklüğünü etmen yoğunluğuna bağlı olarak ele almaktadır. Bu seçeneklerin uygulanması durumunda mızıkçı iletilerin ortaya çıkma olasılığı problem büyüklüğüne göre farklılık gösterecektir. ÖGS mekanizmasının temel amacı geri sarma sayısını azaltmaktır. Dolayısıyla ÖGS ve Zaman Bükülmesi mekanizmalarının karşılaştırmasını yaparken, farklı geri sarma sayılarının ortaya çıkacağı durumları ele almak daha anlamlı sonuçlar görmeyi sağlayacaktır. Böylece hangi durumlarda ÖGS mekanizmasının kayda değer getiriler sağladığının, hangi durumlarda sağlamadığının gösterilmesi amaçlanmıştır. Bu amaçla ÖGS mekanizmasının başarımı farklı etmen yoğunlukları üzerinden incelenmiştir. Yani etmen sayıları farklı değerlerle denenmiş, ancak benzetim uzayının boyutları sabit tutulmuştur.

7.5. Deney Sonuçlarının Değerlendirilmesi

Deneyler sonucunda ÖGS mekanizması ve Zaman Bükülmesi mekanizması şu açılardan karşılaştırılmıştır;

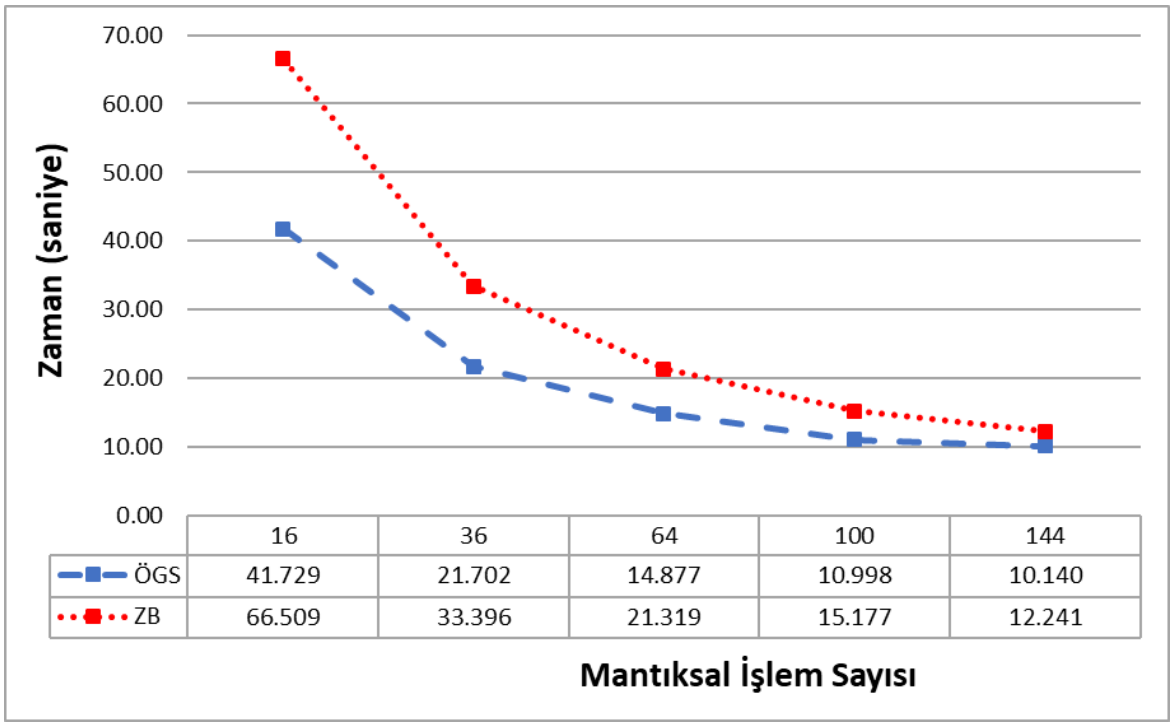
- Benzetimlerin çalışma zamanı,
- Tek işlemciye göre elde edilen hızlanma oranı ve verimlilik,
- Kaçınılan ve çalıştırılmak zorunda kalınan geri sarma işlemleri sayısı,
- Görülen en yüksek denetim noktası aralıkları ve
- Denetim noktasında kaydedilen benzetim durumu boyutları.

Mİ'ler arasındaki etkileşim yoğunluğu iyimser zaman yönetim mekanizmalarının başarımını etkileyen bir faktördür. Etkileşim olasılığının çok olduğu bir benzetimde Mİ'lerin iyimser davranması geri sarma işlemlerinin artmasına yol açarak başarımın düşmesine neden olabilir. Etkileşim olasılığı az olduğunda ise Mİ'lerin yerel zamanlarını diğer Mİ'lerden bağımsız olarak iletmesi gereksiz zaman uyumlama adımlarından kaçınılmasını sağlayarak başarımın yükselmesini sağlayabilir. Uzamsal-koşut benzetimlerde etkileşim olasılığını belirleyen şey Mİ'lerin yönettiği yerel uzayın sınırlarında (tampon alanlarda) yer alan ve komşu Mİ'ler ile etkileşime girilmesine neden olan etmenlerdir. Bir Mİ'nin yönettiği etmen sayısı arttıkça, tampon alanda etmenlerin yer alması olasılığı da artar. Dolayısıyla etmen sayısının artması Mİ'ler arasındaki etkileşim olasılığını da artıracaktır. Bu nedenle, bu çalışmada yapılan deneylerde Zaman Bükülmesi ve ÖGS mekanizmalarının karşılaştırılması farklı etmen yoğunlukları için ayrı ayrı gözlenmiştir.

Yapılan deneyler sonucunda ÖGS mekanizması ile genişletilen Repast HPC (RHPC_PR) ve Zaman Bükülmesi mekanizmasını kullanan Repast HPC'nin (RHPC_TW) düşük ve yüksek etmen yoğunluklu benzetimleri tamamlama süreleri Çizelge 7.1 ve Çizelge 7.2'de verilmiştir. Yapılan her bir deney en az 4 kez tekrarlanmış olup, Çizelge 7.1 ve Çizelge 7.2'de bu deneylerin ortalama değerlerine ve standart sapmalarına yer verilmiştir. Elde edilen sonuçlar Şekil 7.33 ve Şekil 7.34'te verilen grafikler aracılığıyla karşılaştırılabilir. Ayrıca bu algoritmalar kullanıldığında elde edilen hızlanma değerleri Şekil 7.35 ve Şekil 7.36'da; verimlilik değerleri ise Şekil 7.37 ve Şekil 7.38'de gösterilmiştir. Verimlilik değeri, $E = \frac{S}{p}$ eşitliğine göre hesaplanmıştır [121]. Bu eşitlikte E verimlilik değerini; S hızlanma değerini; p ise kullanılan çekirdek sayısını belirtmektedir. Hızlanma değerlerinin ölçülmesi için TRUBA YBH altyapısındaki özdeş çekirdeklerden bir tanesi kullanılmıştır. Ancak adil bir hesaplama olabilmesi için tek çekirdekte çalıştırılan benzetimlerde Repast HPC'nin koşut altyapısı ve zaman uyumlama mekanizması iptal edilmiştir. Yalnızca Repast HPC'nin sağladığı etmen ve ortam veri yapıları ile benzetim motoru için sağladığı altyapılardan yararlanılmıştır.

Çizelge 7.1. Düşük etmen yoğunluklu benzetimlerin farklı sayılardaki Mİ'ler ile çalıştırılma süresi ve standart sapma değerleri

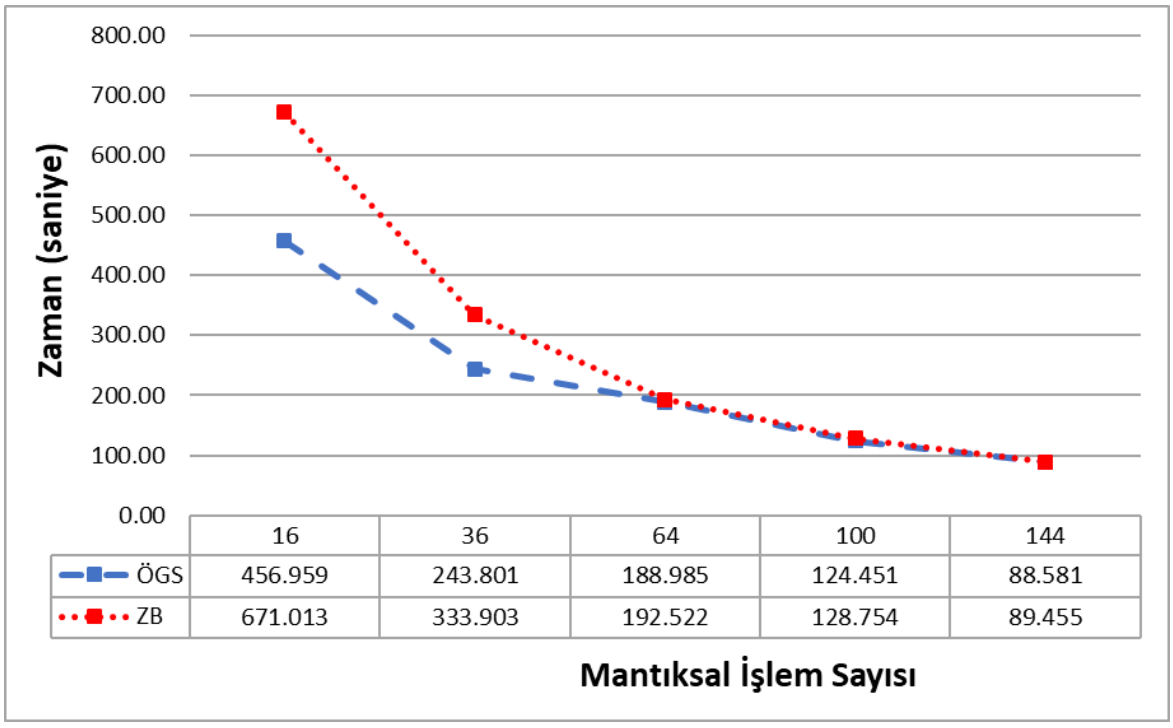
		Mİ sayısı				
		16	**36**	**64**	**100**	**144**
Zaman Bükülmesi	**Geçen zaman [s]**	66,509	33,396	21,319	15,177	12,241
	Standart sapma (σ) [s]	0,454	0,240	0,850	0,106	0,120
ÖGS	**Geçen zaman [s]**	41,729	21,702	14,877	10,998	10,140
	Standart sapma (σ) [s]	1,122	0,605	0,085	0,304	0,458



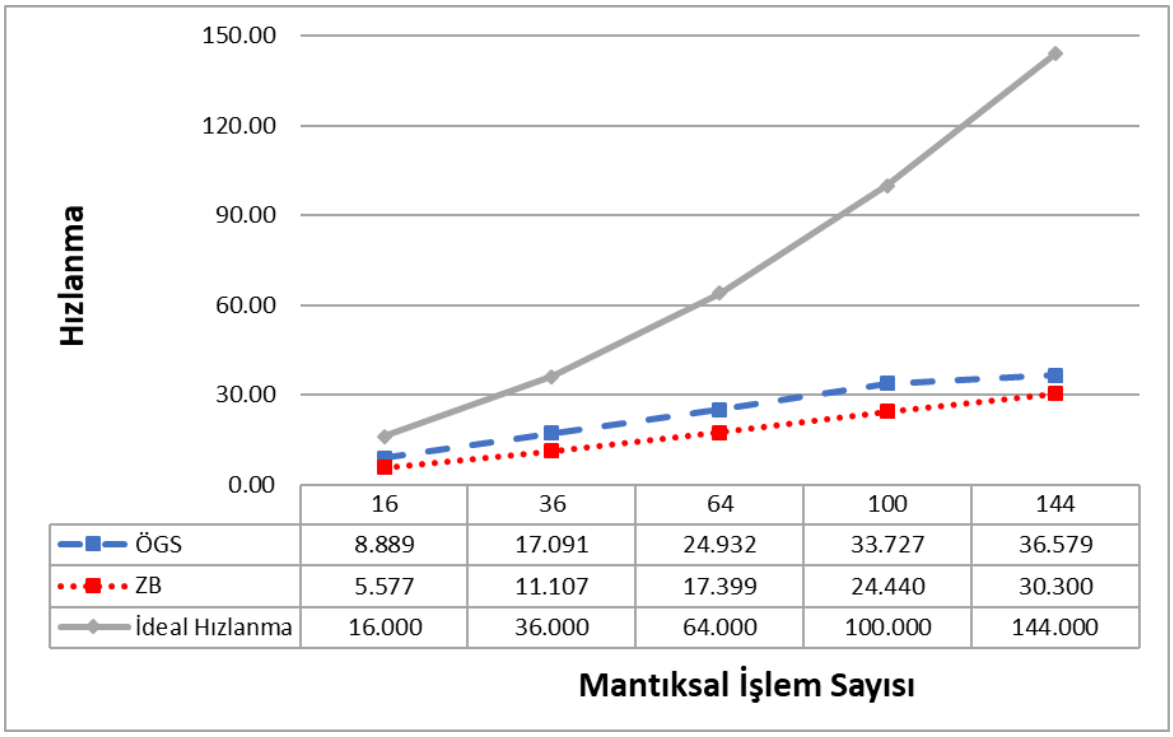
Şekil 7.33. Düşük etmen yoğunluklu benzetimlerin farklı sayılardaki Mİ'ler ile çalıştırılma süresi

Çizelge 7.2. Yüksek etmen yoğunluklu benzetimlerin farklı sayılardaki Mİ'ler ile çalıştırılma süresi ve standart sapma değerleri

		Mİ sayısı				
		16	**36**	**64**	**100**	**144**
Zaman Bükülmesi	**Geçen zaman [s]**	671,01	333,90	192,52	128,75	89,46
	Standart sapma (σ) [s]	17,02	5,00	1,30	8,19	1,31
ÖGS	**Geçen zaman [s]**	456,96	243,80	188,99	124,45	88,58
	Standart sapma (σ) [s]	5,40	13,78	0,96	0,89	1,02

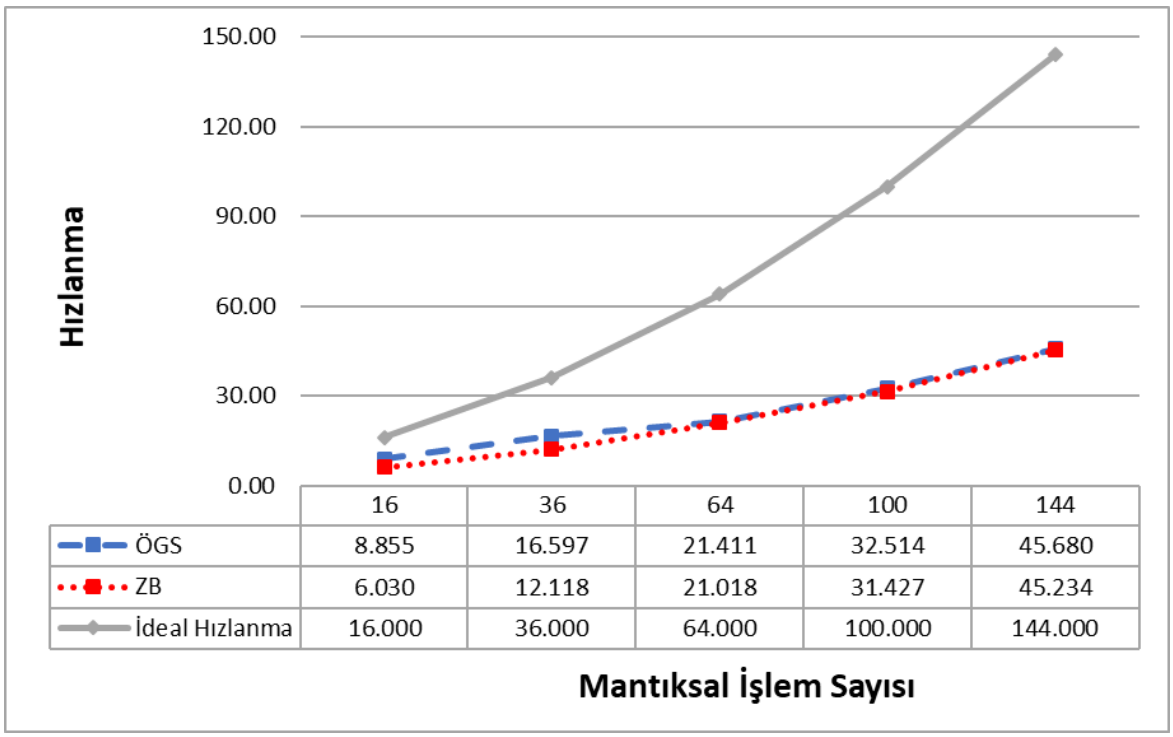


Şekil 7.34. Yüksek etmen yoğunluklu benzetimlerin farklı sayılardaki Mİ'ler ile çalıştırılma süresi



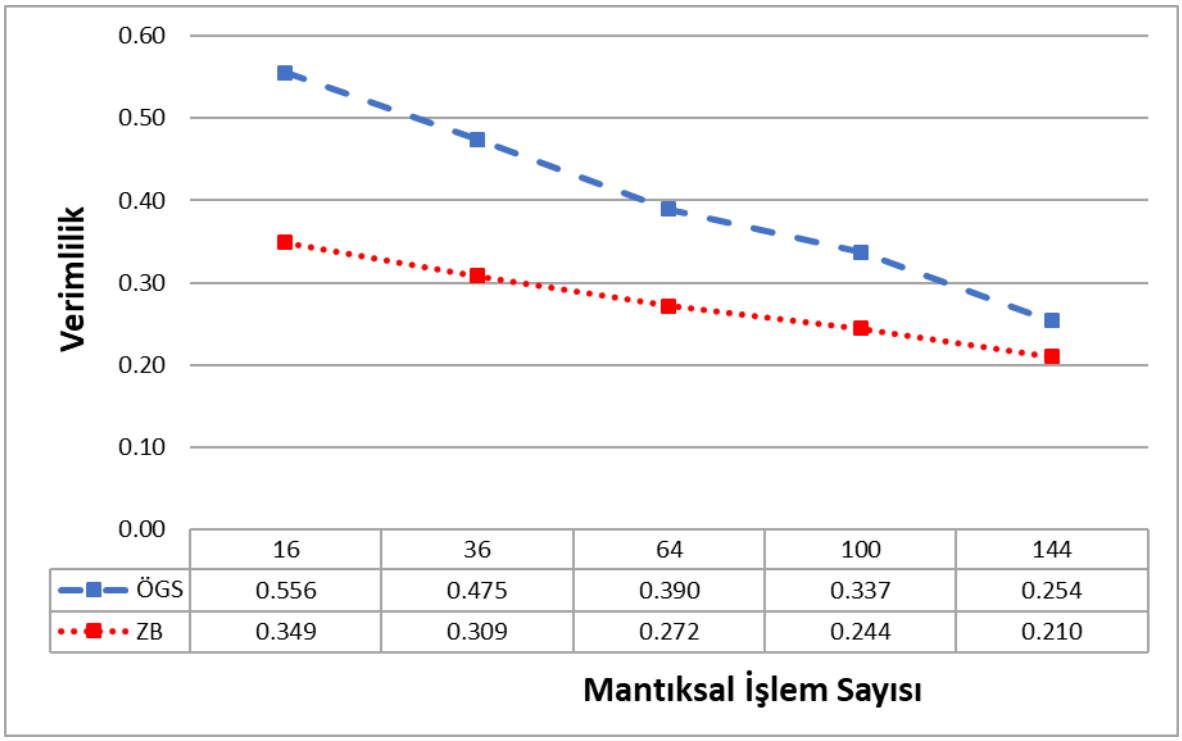
	16	36	64	100	144
ÖGS	8.889	17.091	24.932	33.727	36.579
ZB	5.577	11.107	17.399	24.440	30.300
İdeal Hızlanma	16.000	36.000	64.000	100.000	144.000

Şekil 7.35. Düşük etmen yoğunluklu benzetimlerde elde edilen hızlanma değerleri

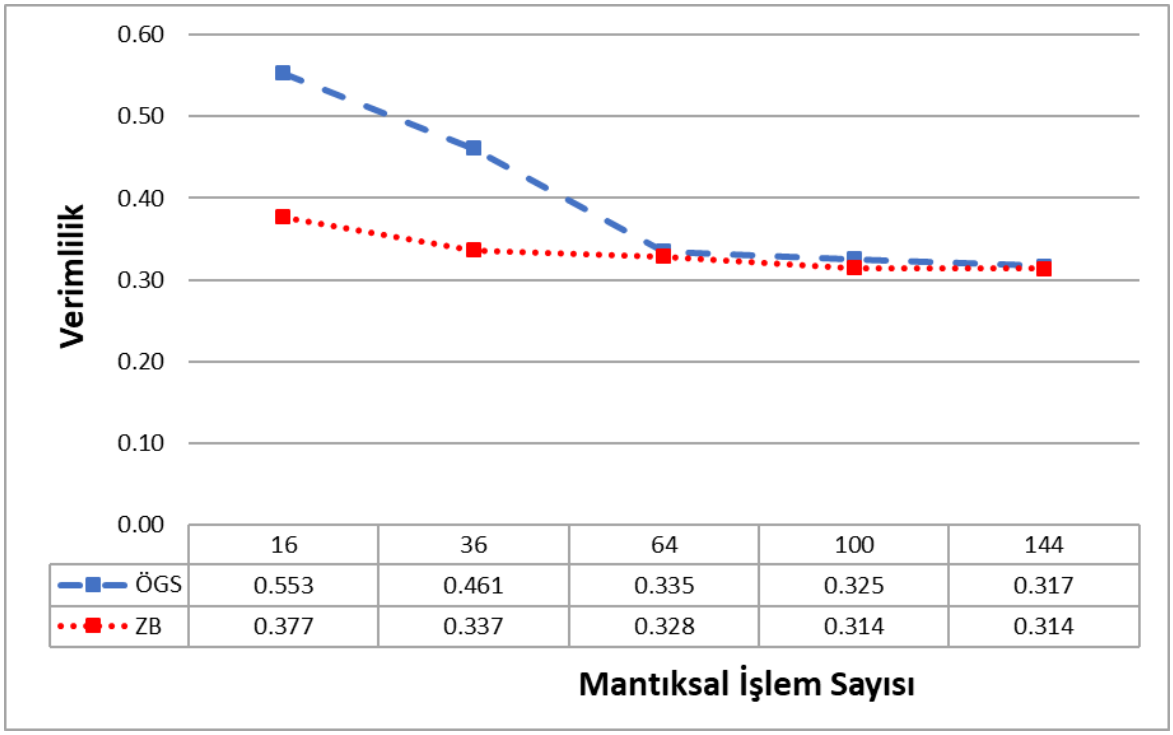


	16	36	64	100	144
ÖGS	8.855	16.597	21.411	32.514	45.680
ZB	6.030	12.118	21.018	31.427	45.234
İdeal Hızlanma	16.000	36.000	64.000	100.000	144.000

Şekil 7.36. Yüksek etmen yoğunluklu benzetimlerde elde edilen hızlanma değerleri



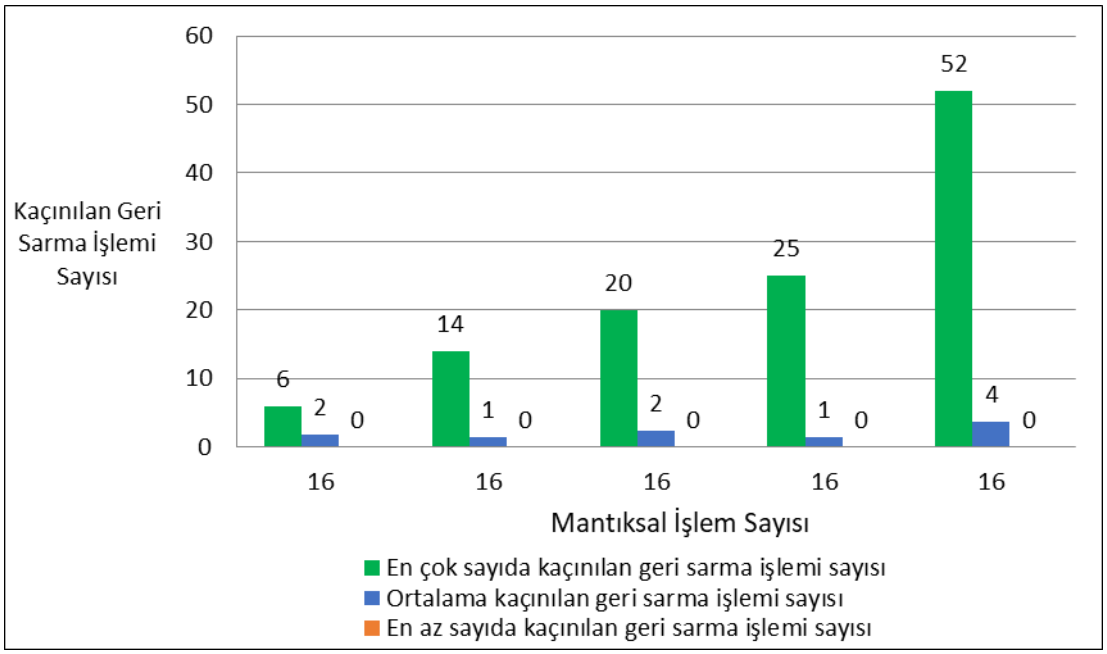
Şekil 7.37. Düşük etmen yoğunluklu benzetimlerde elde edilen verimlilik değerleri



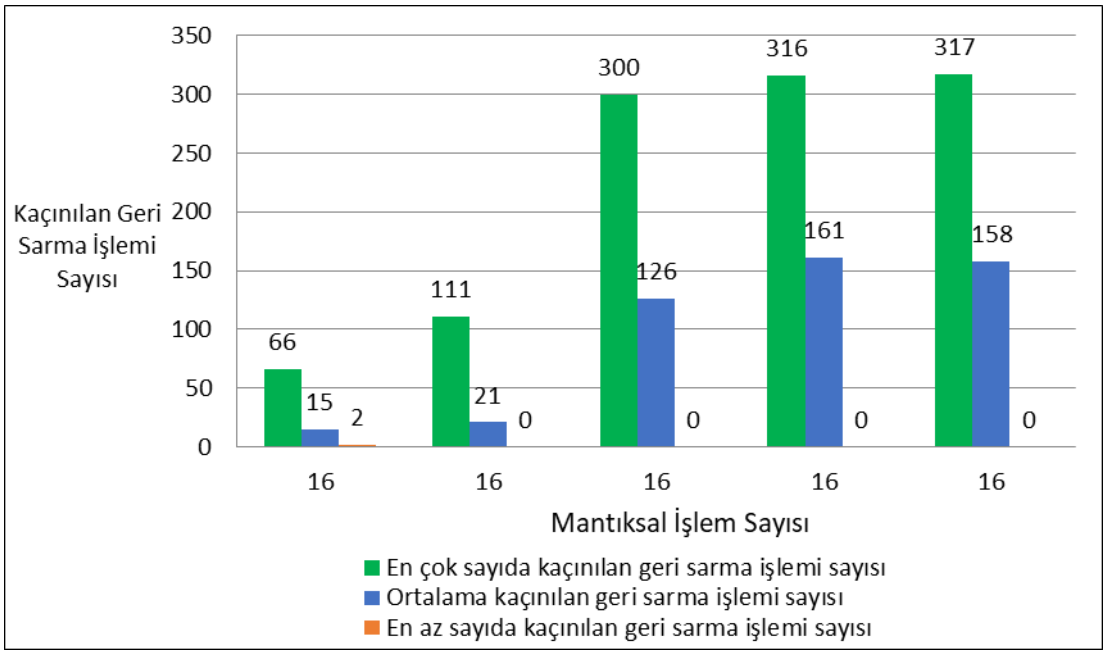
Şekil 7.38. Yüksek etmen yoğunluklu benzetimlerde elde edilen verimlilik değerleri

Elde edilen bu sonuçlara göre ÖGS mekanizmasının Zaman Bükülmesi mekanizmasından daha hızlı çalışabildiği söylenebilir. Buna göre etmen yoğunluğu düşük olarak çalıştırılan tüm deneylerde ÖGS mekanizması Zaman Bükülmesi mekanizmasına kıyasla gözle görülür bir şekilde benzetimlerin daha hızlı tamamlanmasını sağlamıştır. Etmen yoğunluğu yüksek iken ise Mİ sayısı düşük olduğunda yine ÖGS mekanizması gözle görülür bir şekilde daha başarılı olmuştur. Mİ sayısı artırıldığında ÖGS mekanizması daha iyi durumda gözükse de her iki mekanizma arasındaki fark çok düşük seviyelere gelmiştir. Bu değişimin nedeni ise toplamdaki tampon alanı büyüklüğünün artmasına bağlı olarak Mİ'ler arasındaki etkileşim olasılığının artmasıdır. Etkileşim olasılığı arttıkça Mİ'lerin iyimserliği yeterince artamamakta ve her iki yöntem de benzer bir başarıma sahip olmaktadır.

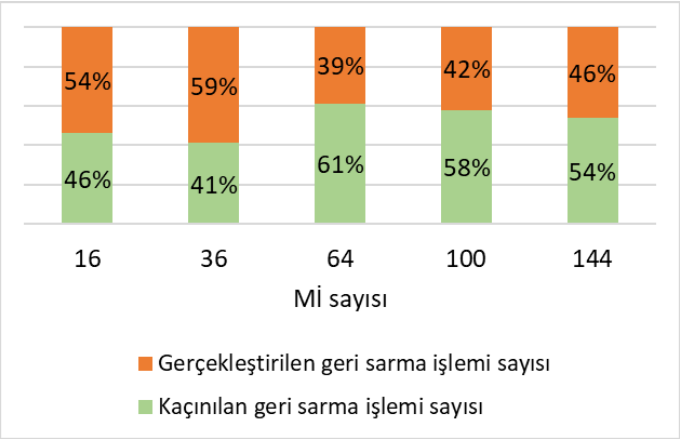
İki zaman yönetim mekanizması arasında bu kadar başarım farkı ortaya çıkmasının nedeni ÖGS mekanizmasının yüksek maliyetli bazı geri sarma işlemlerinden kaçınabilme yeteneğidir. Yukarıdaki deneylerde ÖGS mekanizması sayesinde kaçınılabilen geri sarma işlemlerinin sayısı Şekil 7.39 ve Şekil 7.40'ta düşük ve yüksek yoğunluklu benzetimler için verilmiştir. Bu şekillerde yeşil renk ile gösterilen değerler benzetim boyunca en çok geri sarma işleminden kaçınan Mİ'ye ait kaçınma sayısını; turuncu renk ile gösterilen değerler benzetim boyunca en az geri sarma işleminden kaçınan Mİ'ye ait kaçınma sayısını; mavi renk ile gösterilen değerler ise benzetim boyunca Mİ başına düşen ortalama geri sarma işleminden kaçınma sayısını göstermektedir. Şekil 7.41 ve Şekil 7.42'de ise kaçınılan geri sarma işlem sayısının gerçekleştirilen işlem sayısına oranı verilmiştir. Şekil 7.39 – Şekil 7.42'de verilen grafiklerde dikkat edilmesi gereken nokta şudur: bir Mİ geri sarma işleminden kaçınırken başka bir Mİ geri sarma işlemini gerçekleştirmek zorunda kalmış olabilir. Benzer şekilde bir Mİ, bir komşusundan gelen mızıkçı bir ileti için geri sarma işleminden kaçınırken, diğer bir komşusundan gelen mızıkçı ileti için geri sarma işleminden kaçınamayabilir. Bu nedenle, örneğin Şekil 7.40'ta 64, 100 ve 144 Mİ'nin kullanıldığı benzetimlerde geri sarma işlemlerinden çok defa kaçınılmış gibi gözükmesine rağmen kaçınılamayan geri sarma işlemleri her iki mekanizmanın da Şekil 7.36'daki gibi benzer sonuçlar vermesine neden olmuştur. Şekil 7.39 – Şekil 7.42'de verilen grafiklerden çıkarılacak sonuç, Zaman Bükülmesi mekanizmasındaki Mİ'lerin gerçekleştirmek zorunda kaldığı birçok geri sarma işleminin aslında gereksiz olduğu ve bunlardan kaçınılabileceğidir.



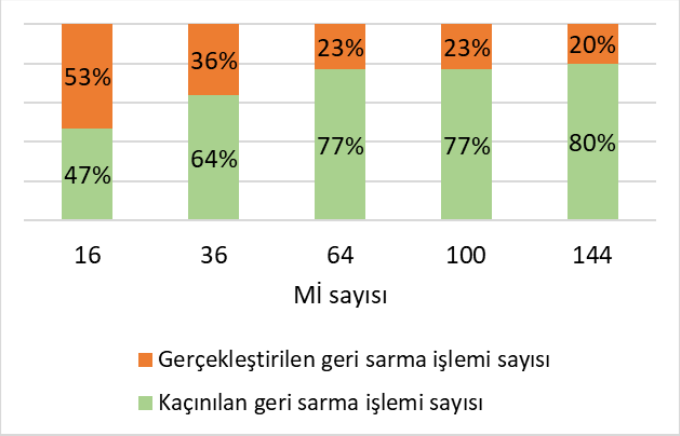
Şekil 7.39. Etmen yoğunluğunun düşük olduğu deneylerde kaçınılan geri sarma işlemi sayısı



Şekil 7.40. Etmen yoğunluğunun yüksek olduğu deneylerde kaçınılan geri sarma işlemi sayısı



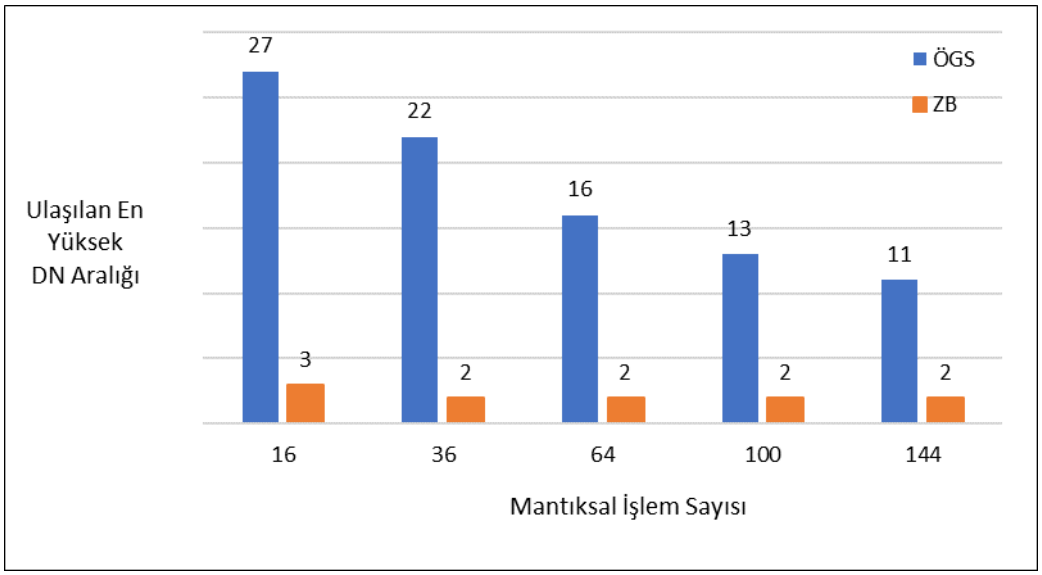
Şekil 7.41. Etmen yoğunluğunun düşük olduğu deneylerde kaçınılan ve gerçekleştirilen geri sarma işlemlerinin yüzde olarak karşılaştırılması



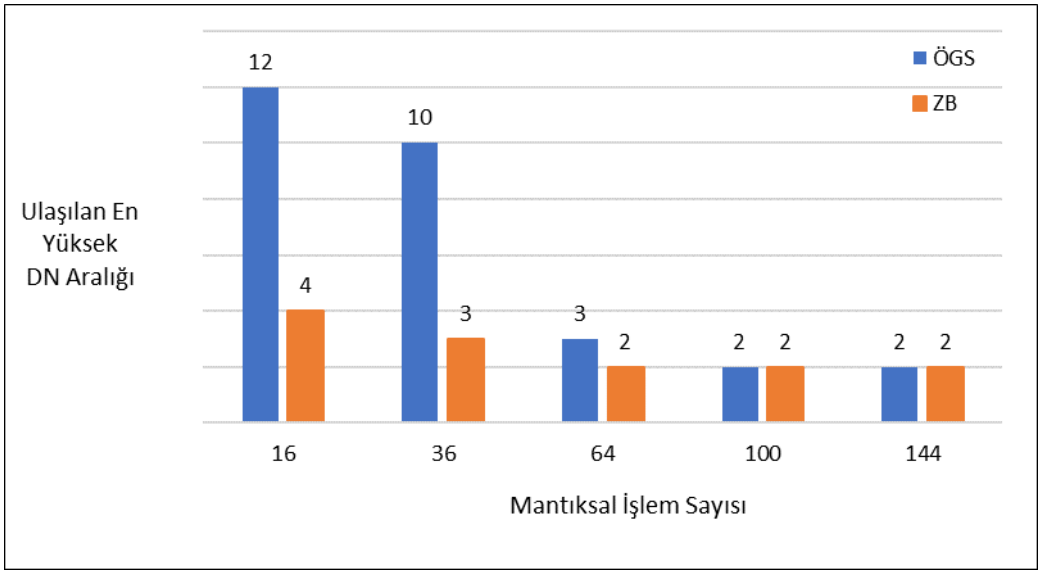
Şekil 7.42. Etmen yoğunluğunun yüksek olduğu deneylerde kaçınılan ve gerçekleştirilen geri sarma işlemlerinin yüzde olarak karşılaştırılması

Hem ÖGS, hem de Zaman Bükülmesi mekanizmaları denetim noktası sıklığını Bölüm 5.3.2'de ayrıntıları verilen TAÇA algoritmasına göre belirlemektedir. Bu algoritmaya göre geri sarma işlemi gerçekleşmediği sürece denetim noktası sıklığı artırılmaktadır. Dolayısıyla gereksiz geri sarma işlemlerinin tespit edilip bu geri sarma işlemlerinden kaçınılması denetim noktalarının daha seyrek alınmasını sağlayacaktır. Bu nedenle hem ÖGS hem de Zaman Bükülmesi mekanizmalarının benzetim boyunca ulaştığı en yüksek denetim noktası aralıkları gözlenmiş olup elde edilen sonuçlar Şekil 7.43 ve Şekil 7.44'te verilmiştir. Bu grafiklerden görüleceği üzere ÖGS mekanizması hemen hemen tüm deneylerde daha seyrek denetim noktası almıştır. Çünkü ÖGS mekanizması birçok geri sarma işleminden kaçınmayı sağlamış ve Mİ'lerin daha iyimser davranabilmesine izin vermiştir. Şekil 7.44'te

görülen ve Mİ sayısının 64, 100, 144 olduğu deneylerde ise Mİ'ler arası etkileşim arttığı için ÖGS mekanizması da çok sayıda geri sarma işlemiyle karşılaşmış ve denetim noktası alma aralığını artıramamıştır. Bu durum, ÖGS mekanizmasının Zaman Bükülmesi mekanizmasına göre herhangi bir avantaj sağlayamamasına ve her iki mekanizmanın da Şekil 7.34'te görüldüğü gibi birbirine çok yakın sonuçlar vermesine neden olmuştur.



Şekil 7.43. Etmen yoğunluğunun düşük olduğu deneylerde gözlenen en yüksek denetim noktası aralıkları



Şekil 7.44. Etmen yoğunluğunun yüksek olduğu deneylerde gözlenen en yüksek denetim noktası aralıkları

Elde edilen sonuçların bir özetini vermek gerekirse;

- Mİ'ler arasındaki etkileşim çok sık olmadığında ÖGS mekanizması daha iyimser davranabilmekte ve benzetimleri Zaman Bükülmesi mekanizmasından çok daha hızlı tamamlayabilmektedir. Mİ'ler arası etkileşim olasılığı arttıkça ÖGS mekanizmasının da bu üstünlüğü kaybetmeye başladığı görülmektedir. Etkileşim olasılığının çok artması durumunda zaten sakıngan yaklaşımların iyimser yaklaşımlara göre daha tercih edilebilir olduğu bilinmektedir [131]. Bu nedenle ÖGS mekanizmasının Zaman Bükülmesi mekanizmasına göre daha iyi bir başarıma sahip olduğu ve daha tercih edilebilir olduğu söylenebilir.
- ÖGS mekanizmasının başarımının Mİ'ler arasındaki etkileşim olasılığına bağlı olduğu görülmektedir. Etkileşim olasılığının da etmen yoğunluğu ve tampon alanı büyüklüğünden etkilendiği söylenebilir.
- ÖGS mekanizması çok sayıdaki geri sarma işlemini kaçınılabilir olarak sınıflandırmış ve bu geri sarma işlemlerini çalıştırmamıştır. Bu durum benzetimlerin doğruluğunu ve tekrarlanabilirliğini bozmamıştır; çünkü bu geri sarma işlemlerinin nedenini oluşturan mızıkçı iletiler benzetime dahil edilse de edilmese de benzetimin sonucu aynı olacaktır.
- ÖGS mekanizmasının kullanıldığı deneylerde elde edilen en yüksek denetim noktası aralığı, Zaman Bükülmesi mekanizmasının kullanıldığı deneylerde elde edilen en yüksek denetim noktası aralığından büyük veya eşit olarak ölçülmüştür. Çünkü ÖGS mekanizmasının gereksiz geri sarma işlemlerinden kaçınması, TAÇA algoritmasının denetim noktası aralıklarını artırmasına neden olmaktadır. Bu da Mİ'lerin daha iyimser davranmasına izin verilmesini sağlamaktadır. Bu nedenle ÖGS mekanizması Zaman Bükülmesi mekanizmasına göre daha koşut çalışabilmektedir.

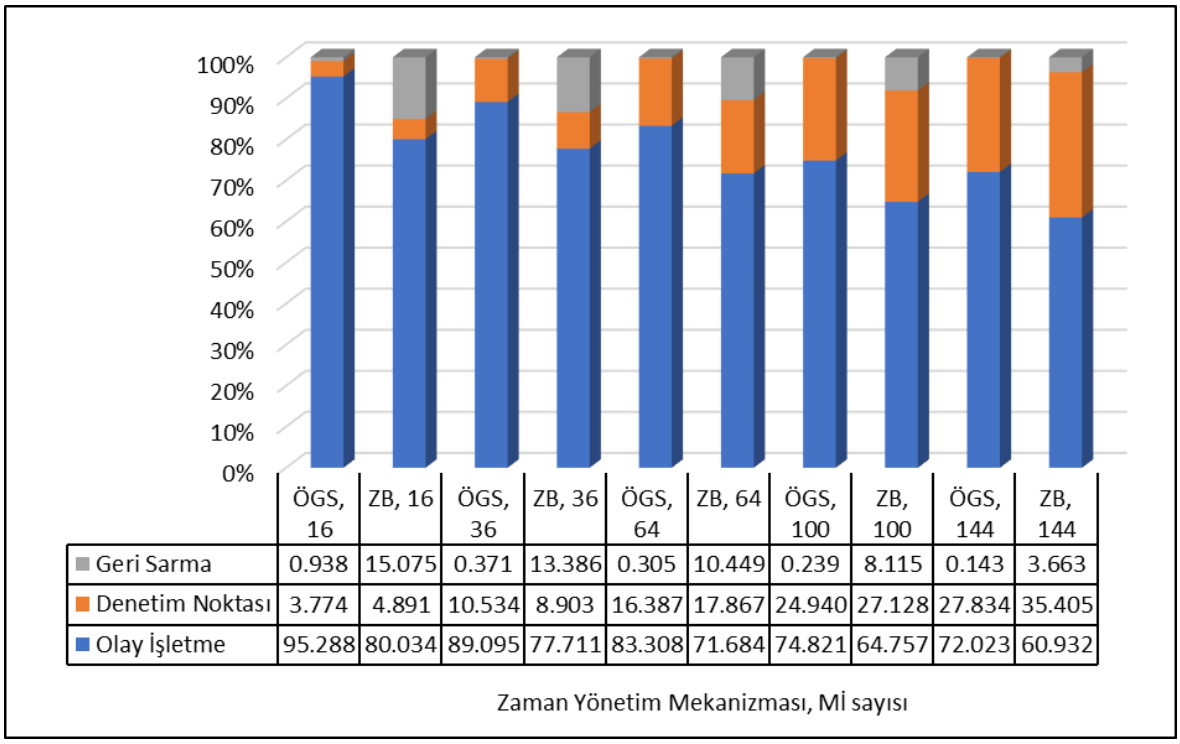
ÖGS mekanizması ve bu mekanizmanın kullanılmasıyla elde edilen deney sonuçları [132]'de sunulmuştur.

7.5.1. Zaman Yönetim Mekanizmalarının Çalışma Tutumları

Gerçekleştirilen deneylerde benzetimlerin sahip olduğu çalışma tutumları (İng. *profile*) Şekil 7.45 ve Şekil 7.46'da gösterilmiştir. Bu şekillerde; geri sarma, durum

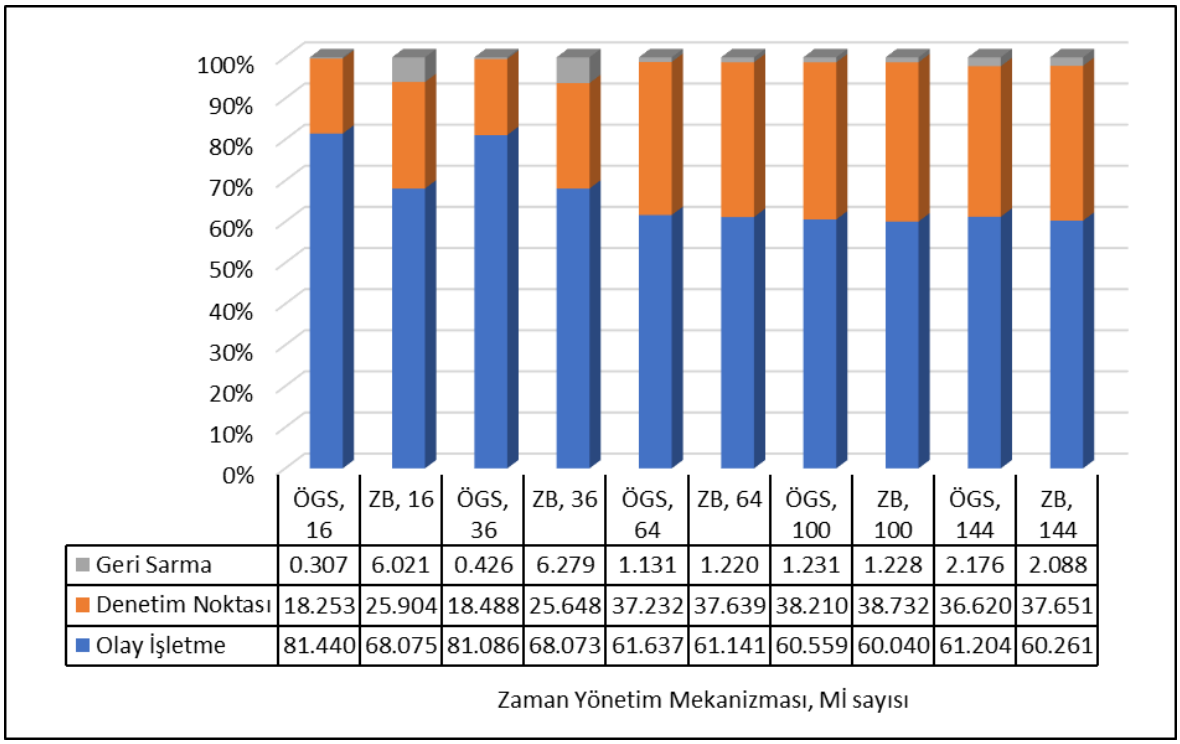
geri yükleme (Şekil 3.1'de adım 1), ileri sarma (Şekil 3.1'de adım 2) ve mızıkçı iletiyi işletme maliyetini (Şekil 3.1'de adım 3); denetim noktası, durum kaydetme ve zaman uyumlama için gerekli bekleme zamanını; olay işletme ise benzetim olaylarının işletilmesini ve ileri sarma sonrası yeniden çalıştırılan olayların işletilmesini (Şekil 3.1'de adım 4) içermektedir. ÖGS mekanizmasıyla birlikte gelen etkileşim maskesi yaratma ve maskeleme işlemlerinden kaynaklı ortaya çıkan maliyetlerin tüm deneyler için toplam çalışma süresinin %0,1'inden daha az zaman aldığı görülmüştür. Dolayısıyla Şekil 7.45 ve Şekil 7.46'da bu maliyetlere yer verilmemiştir.

Ortaya çıkan bu çalışma tutumları incelendiğinde, Zaman Bükülmesi mekanizmasının ÖGS mekanizmasına göre geri sarma işlemlerine daha çok zaman ayırdığı görülmüştür. Bu durumun temel nedeni Zaman Bükülmesi mekanizmasının ÖGS mekanizmasından daha fazla sayıda geri sarma işlemi yapmak zorunda kalmasıdır. Ayrıca kullanılan Mİ sayısı arttıkça etkileşim olasılığı da arttığından dolayı hem Zaman Bükülmesi hem de ÖGS mekanizmasının çalışma tutumları benzer eğilimler göstermeye başlamaktadır (Şekil 7.46). Şekil 7.44'te görüldüğü üzere her iki zaman yönetim mekanizması da yüksek sayıda Mİ kullanıldığında denetim noktası sıklığını çok artıramadığından dolayı, her iki mekanizmanın da denetim noktası için ayırdığı zaman Mİ sayısı ile doğru orantılı olarak artmaktadır.



	ÖGS, 16	ZB, 16	ÖGS, 36	ZB, 36	ÖGS, 64	ZB, 64	ÖGS, 100	ZB, 100	ÖGS, 144	ZB, 144
Geri Sarma	0.938	15.075	0.371	13.386	0.305	10.449	0.239	8.115	0.143	3.663
Denetim Noktası	3.774	4.891	10.534	8.903	16.387	17.867	24.940	27.128	27.834	35.405
Olay İşletme	95.288	80.034	89.095	77.711	83.308	71.684	74.821	64.757	72.023	60.932

Şekil 7.45. Düşük etmen yoğunluklu benzetimlerin çalışma tutumları



	ÖGS, 16	ZB, 16	ÖGS, 36	ZB, 36	ÖGS, 64	ZB, 64	ÖGS, 100	ZB, 100	ÖGS, 144	ZB, 144
Geri Sarma	0.307	6.021	0.426	6.279	1.131	1.220	1.231	1.228	2.176	2.088
Denetim Noktası	18.253	25.904	18.488	25.648	37.232	37.639	38.210	38.732	36.620	37.651
Olay İşletme	81.440	68.075	81.086	68.073	61.637	61.141	60.559	60.040	61.204	60.261

Şekil 7.46. Yüksek etmen yoğunluklu benzetimlerin çalışma tutumları

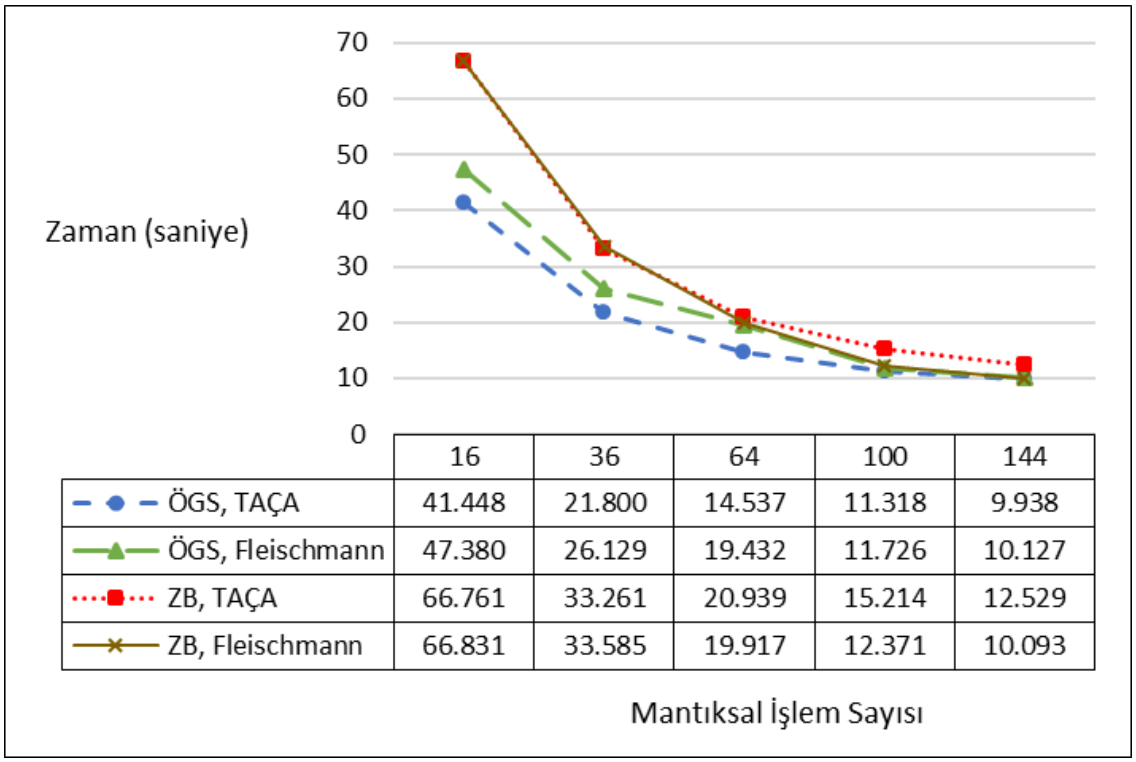
7.5.2. Denetim Noktası Mekanizmalarının Karşılaştırılması

ÖGS mekanizmasının, Zaman Bükülmesi mekanizması ile karşılaştırılmasında denetim noktası sıklığı için TAÇA yöntemi kullanılmıştır. TAÇA yönteminin karşılaştırılan mekanizmalardan herhangi birisine avantaj sağlayıp sağlamadığını kontrol etmek amacıyla literatürde bulunan farklı bir denetim noktası sıklığı belirleme mekanizması ile de deneyler tekrarlanmıştır. Literatürdeki yöntemlerden bir çoğu geri sarma işlemleri ve denetim noktası alma maliyetlerini göz önünde bulundurarak denetim noktasının ne zaman alınacağını belirler. Fleischmann tarafından önerilen yöntem ileri sarma maliyeti ve durum kaydetme maliyetini göz önünde bulundurarak denetim noktası sıklığını artırır veya azaltır [43]. Fleischmann yönteminin tercih edilme nedenleri arasında; literatürdeki çalışmalarda sıkça kullanılması, geri sarma ve denetim noktası alma maliyetleri arasındaki dengeyi gözeten bir yöntem olması ve kendisinden sonra ortaya çıkmış ve dikkate değer bir şekilde kendisini geçen bir çalışmaya rastlanılamaması sayılabilir.

Tekrarlanan deneyler aynı durum çalışması için aynı şartlar altında gerçekleştirilmiştir. Bu deneyler için düşük ve yüksek etmen yoğunluklu benzetimlerin tamamlanma süreleri Çizelge 7.3 ve Çizelge 7.4'te verilmiştir. Yapılan her bir deney en az 4 kez tekrarlanmış olup, Çizelge 7.3 ve Çizelge 7.4'te bu deneylerin ortalama değerlerine ve standart sapmalarına yer verilmiştir. Bu deneylerde elde edilen benzetim tamamlanma süreleri ve hızlanma değerleri Şekil 7.47 – Şekil 7.50'de verilen grafiklerden görülebilir. Elde edilen sonuçlara göre denetim noktası sıklığını belirleyen mekanizmaların, hem ÖGS hem de Zaman Bükülmesi mekanizmalarını etkilediği, ancak birinin diğerinin önüne geçebilecek kadar bir etkiye sahip olmadığı görülmüştür.

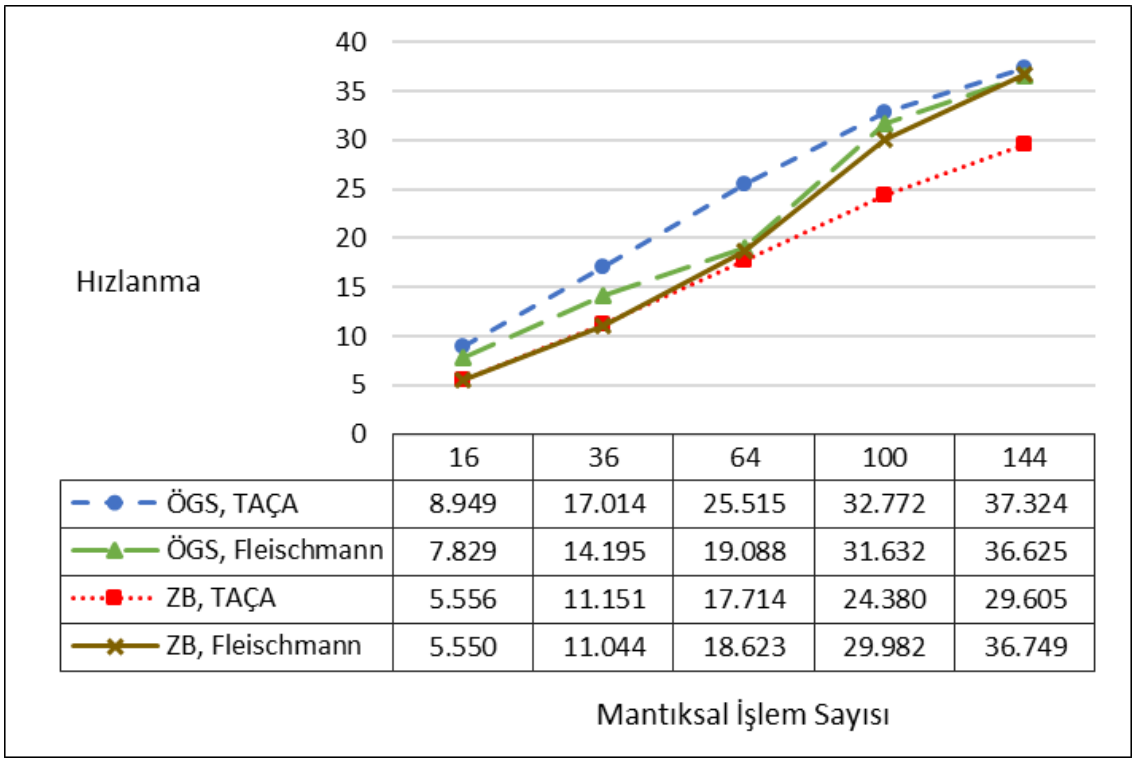
Çizelge 7.3. Düşük etmen yoğunluklu benzetimlerin farklı sayılardaki Mİ'ler ve farklı denetim noktası mekanizmaları ile çalıştırılma süresi ve standart sapma değerleri

		Mİ sayısı				
		16	**36**	**64**	**100**	**144**
Zaman Bükülmesi (TAÇA)	**Geçen zaman [s]**	41.448	21.800	14.537	11.318	9.938
	Standart sapma (σ) [s]	1.484	0.959	0.410	0.439	0.162
Zaman Bükülmesi (Fleischmann)	**Geçen zaman [s]**	47.380	26.129	19.432	11.726	10.127
	Standart sapma (σ) [s]	2.120	3.581	0.413	0.918	0.866
Zaman Bükülmesi (TAÇA)	**Geçen zaman [s]**	66.761	33.261	20.939	15.214	12.529
	Standart sapma (σ) [s]	0.974	0.401	0.561	0.416	0.271
Zaman Bükülmesi (Fleischmann)	**Geçen zaman [s]**	66.831	33.585	19.917	12.371	10.093
	Standart sapma (σ) [s]	0.796	0.601	1.673	1.134	0.419



	16	36	64	100	144
ÖGS, TAÇA	41.448	21.800	14.537	11.318	9.938
ÖGS, Fleischmann	47.380	26.129	19.432	11.726	10.127
ZB, TAÇA	66.761	33.261	20.939	15.214	12.529
ZB, Fleischmann	66.831	33.585	19.917	12.371	10.093

Şekil 7.47. Düşük etmen yoğunluklu benzetimlerin farklı denetim noktası mekanizmaları ve farklı sayılardaki Mİ'ler ile çalıştırılma süresi

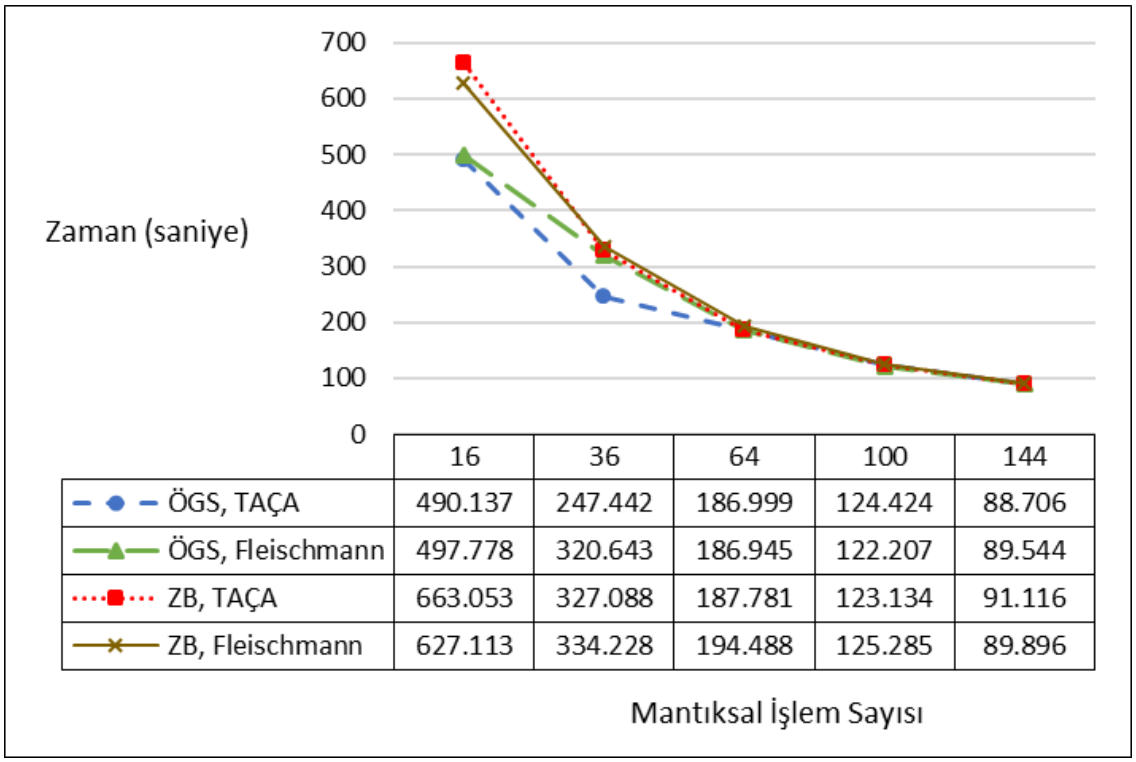


	16	36	64	100	144
ÖGS, TAÇA	8.949	17.014	25.515	32.772	37.324
ÖGS, Fleischmann	7.829	14.195	19.088	31.632	36.625
ZB, TAÇA	5.556	11.151	17.714	24.380	29.605
ZB, Fleischmann	5.550	11.044	18.623	29.982	36.749

Şekil 7.48. Düşük etmen yoğunluklu benzetimler için farklı denetim noktası mekanizmaları ve farklı sayılardaki Mİ'ler ile elde edilen hızlanma değerleri

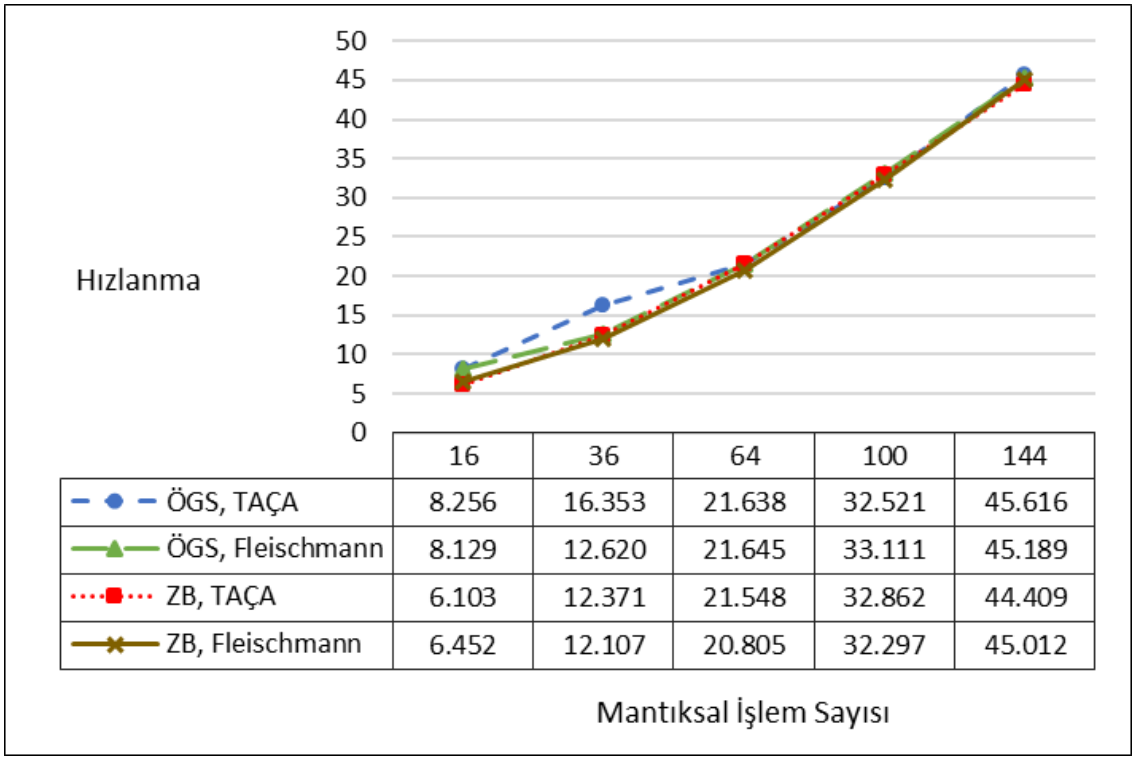
Çizelge 7.4. Yüksek etmen yoğunluklu benzetimlerin farklı sayılardaki Mİ'ler ve farklı denetim noktası mekanizmaları ile çalıştırılma süresi ve standart sapma değerleri

		Mİ sayısı				
		16	**36**	**64**	**100**	**144**
Zaman Bükülmesi (TAÇA)	**Geçen zaman [s]**	490.14	247.44	187.00	124.42	88.71
	Standart sapma (σ) [s]	33.35	16.66	1.60	1.60	1.94
Zaman Bükülmesi (Fleischmann)	**Geçen zaman [s]**	497.78	320.64	186.95	122.21	89.54
	Standart sapma (σ) [s]	58.81	10.41	0.73	0.81	2.72
Zaman Bükülmesi (TAÇA)	**Geçen zaman [s]**	663.05	327.09	187.78	123.13	91.12
	Standart sapma (σ) [s]	24.62	3.09	1.57	2.12	5.99
Zaman Bükülmesi (Fleischmann)	**Geçen zaman [s]**	627.11	334.23	19.49	125.29	89.90
	Standart sapma (σ) [s]	104.17	3.22	1.75	2.16	0.65



	16	36	64	100	144
ÖGS, TAÇA	490.137	247.442	186.999	124.424	88.706
ÖGS, Fleischmann	497.778	320.643	186.945	122.207	89.544
ZB, TAÇA	663.053	327.088	187.781	123.134	91.116
ZB, Fleischmann	627.113	334.228	194.488	125.285	89.896

Şekil 7.49. Yüksek etmen yoğunluklu benzetimlerin farklı denetim noktası mekanizmaları ve farklı sayılardaki Mİ'ler ile çalıştırılma süresi



	16	36	64	100	144
ÖGS, TAÇA	8.256	16.353	21.638	32.521	45.616
ÖGS, Fleischmann	8.129	12.620	21.645	33.111	45.189
ZB, TAÇA	6.103	12.371	21.548	32.862	44.409
ZB, Fleischmann	6.452	12.107	20.805	32.297	45.012

Şekil 7.50. Yüksek etmen yoğunluklu benzetimler için farklı denetim noktası mekanizmaları ve farklı sayılardaki Mİ'ler ile elde edilen hızlanma değerleri

7.5.3. Benzetim Durumu Boyutunun Zaman Yönetim Mekanizmalarına Etkisi

Denetim noktası sırasında kaydedilen benzetim durumları, ilgili Mİ'nin yönettiği etmenlerin ve uzayın bilgilerini içerir. Bu tez çalışması kapsamında yapılan çalışmalarda uzay ile ilgili herhangi bir bilgi kaydedilmemiştir. Bir Mİ, kendi yönettiği etmenlere ek olarak yerel olmayan ancak o an için kendi sınırlarına yakın alanlarda bulunan etmenlerin (komşu Mİ'lerin tampon alanında bulunan etmenler) bilgisini de benzetim durumu içerisinde kaydeder. Bir Mİ'nin $N \times N$ hücreyi yönettiği ve komşuluk alanı büyüklüğünün b olduğu bir uzayda bir benzetim durumu kaydedilirken ortalama olarak $A \cdot \frac{(N+2b)(N+2b)}{L \cdot N^2} \cdot S$ adet tam sayı kaydedilecektir. Burada A benzetimdeki toplam etmen sayısını, S bir etmenin durum değişkenlerinin toplam boyutunu, L ise benzetimi yöneten toplam Mİ sayısını belirtmektedir. Kullanılan modelde her bir etmenin özniteliklerinden 7 tanesi tam sayı (`id`, `startingRank`, `agentType`, `currentRank`, `score`, `jailTerm`, `maskKey` (ÖGS mekanizmasındaki etkileşim modeli için kullanılan asal sayı)), 4 tanesi gerçek sayı (`riskAversion`, `perceivedHardship`, `grievance`, `arrestProbability`) ve 1 tanesi de ikili değer (etmenin bir önceki zaman adımında güncellenip güncellenmediği bilgisi) olarak tutulmaktadır. Ayrıca her bir etmen için Repast HPC 6 tam sayı kullanarak konum bilgisini tutmaktadır. Bu 6 tam sayının 4'ü AgentId bilgisini (bkz. Ek - 3), 2'si ise iki boyutlu uzaydaki konum bilgisini temsil etmektedir. Bu durumda benzetim durumunu kaydetmek için her bir etmen başına toplam 13 tam sayı, 4 gerçek sayı, 1 de ikili değer kaydedilmektedir. Repast HPC'yi derlemek için kullanılan GCC derleyicisi bir tam sayı için 4 byte, bir gerçek sayı için 8 byte ve bir ikili değer için 1 byte ayırmaktadır. Dolayısıyla bir etmenin durum değişkenlerinin kaydedilmesi 85 byte'lık bir alan gerektirmektedir. Yapılan deneyler göz önünde bulundurulduğunda, örneğin 16 Mİ'nin olduğu düşük etmen yoğunluklu bir benzetimde, bir denetim noktası sırasında bir Mİ ortalama olarak $12960 \cdot \frac{902 \cdot 902}{16 \cdot 900^2} \cdot 85 \cong 67{,}5$ kilobyte'lık bir alana ihtiyaç duymaktadır. Benzer şekilde 144 Mİ'nin kullanıldığı yüksek etmen yoğunluklu bir benzetimde ise bu değer yaklaşık olarak 75,7 kilobyte olacaktır.

Benzetim durum büyüklüğünün zaman yönetim mekanizmasının başarımına etkisini incelemek amacıyla etmenlerin durum büyüklüğü yapay olarak 80 katına çıkarılmıştır. Hem Zaman Bükülmesi hem de ÖGS mekanizmaları bu değişimden

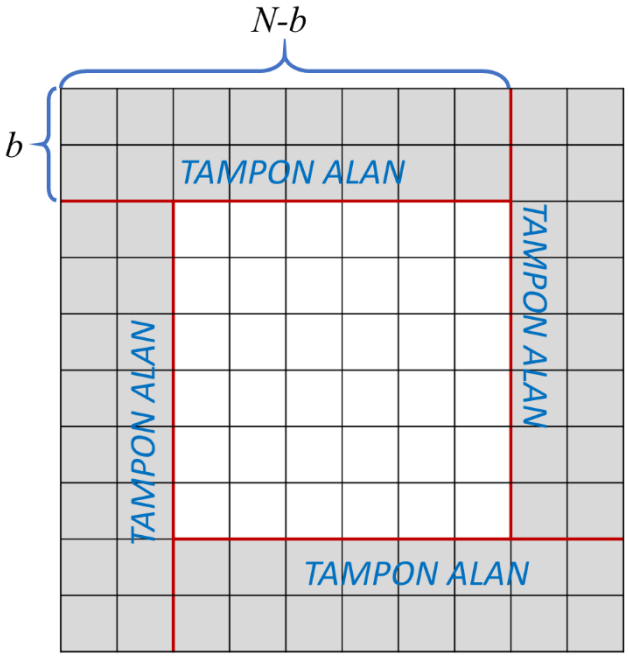
benzer şekilde olumsuz etkilenmiştir. Örneğin 36 Mİ'nin kullanıldığı düşük etmen yoğunluklu bir benzetimde ÖGS mekanizmasının hızlanma değerinin 25,52'den 22,08'e; Zaman Bükülmesi'nin hızlanma değerinin ise 17,71'den 15,83'e düştüğü gözlenmiştir.

7.5.4. Ek iş yükü

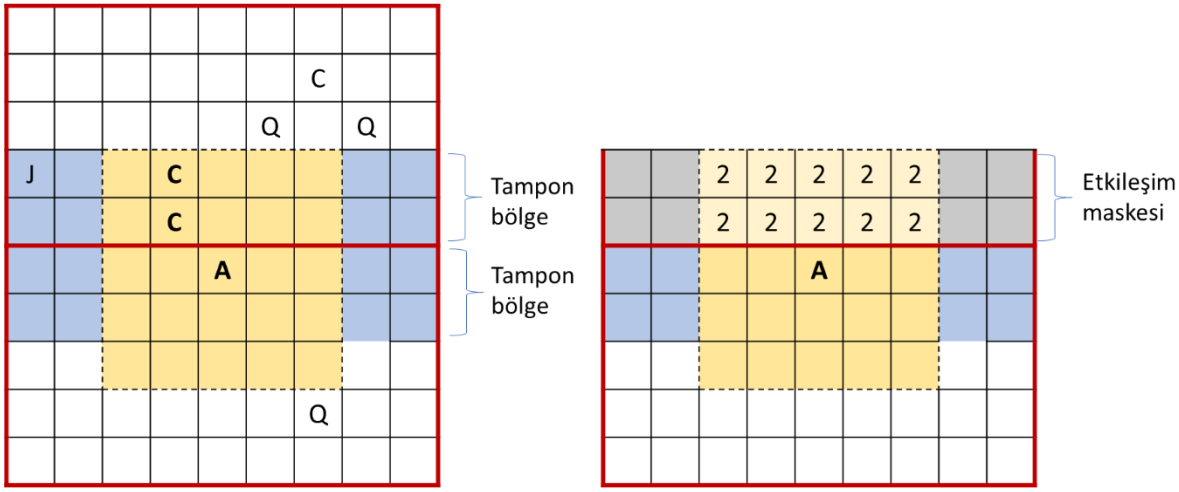
ÖGS mekanizması, Zaman Bükülmesi mekanizmasının iş yüküne ek olarak bazı yeni iş yükleriyle birlikte gelmektedir. Bu iş yükleri etkileşim maskelerinin ve mızıkçı ileti izlerinin oluşturulması sırasında ve maskeleme işlemi sırasında ortaya çıkmaktadır. Ortaya çıkan bu iş yükleri bellek ve işlemci gücü açısından incelenmiştir.

Bir Mİ herhangi bir t anında tampon alanında etmen bulunduruyorsa, bir etkileşim maskesi yaratır. ÖGS mekanizmasının getirdiği ek bellek kullanım ihtiyacı tampon alanında bulunan etmen sayısına bağlıdır. Şekil 7.51'de $N \times N$ hücrenin olduğu ve komşuluk alanı büyüklüğünün b olduğu bir uzayda tampon alanların toplam büyüklüğü gösterilmiştir. Böyle bir uzayda toplam $4b \cdot (N - b)$ adet hücre tampon bölgelerini oluşturmaktadır. Bir Mİ'nin A adet etmeni yönettiği varsayılırsa, herhangi bir t anında tampon bölgede ortalama $A \cdot \frac{4b \cdot (N-b)}{N^2}$ adet etmen bulunacaktır. Bölüm 7.2'de anlatıldığı üzere tampon alandaki her bir etmen için etkileşim maskesindeki belirli noktalara bir asal sayı yerleştirilir. Örneğin komşuluk alanı büyüklüğünün 2 olduğu bir örnekte bir etmen için en çok 10 asal sayı etkileşim maskesine Şekil 7.52'de olduğu gibi yerleştirilir. Daha genel haliyle söylemek gerekirse, bir etmen için en fazla $b \cdot (2b + 1)$ adet değer etkileşim maskesinde yer alır. Tampon bölgede bulunan ortalama etmen sayısı düşünüldüğünde ise bir etkileşim maskesi herhangi bir t anında ortalama olarak $A \cdot \frac{4b \cdot (N-b)}{N^2} \cdot b \cdot (2b + 1)$ adet asal sayı içerir. Eğer tampon bölgedeki etmenler birbirinin komşuluk alanı içerisindeyse, bu etmenlerin kesişimini ifade eden tek bir değer etkileşim maskesinde yer alacağı için bu sayı daha da düşük olacaktır. Şekil 7.53 bu durumu örneklemekte olup iki farklı etmen türü için etkileşim maskesine yerleştirilen değerler farklı renklerle gösterilmiştir. Oluşturulan etkileşim maskeleri ileride bir mızıkçı ileti gelmesi durumuna karşı bir sonraki denetim noktası alınana kadar silinmezler. Dolayısıyla bir denetim noktası aralığı boyunca etkileşim maskelerinin benzetim zamanında gerektireceği bellek

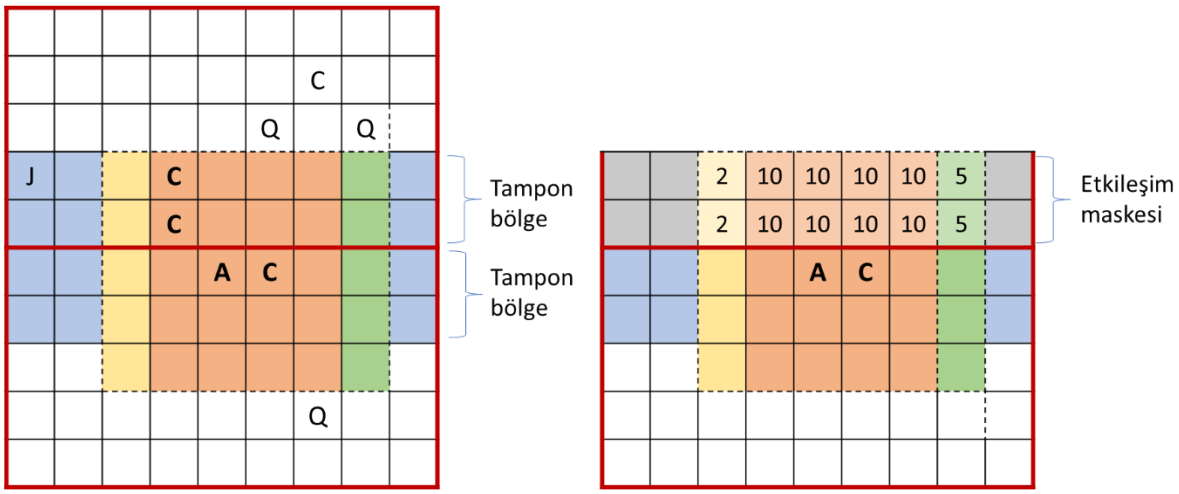
alanı maliyeti $\chi_t \cdot A \cdot \frac{4b \cdot (N-b)}{N^2} \cdot b \cdot (2b+1)$ olarak ifade edilebilir. Bu ifadedeki χ_t denetim noktası aralığını temsil etmektedir.



Şekil 7.51. Her bir Mİ'nin NxN'lik bir ızgara ortamını yönettiği ve komşuluk alanı büyüklüğünün b olduğu bir benzetimin görünümü



Şekil 7.52. Civil Violence modelinde komşuluk alanı büyüklüğü 2 iken bir etkileşim maskesinin içerdiği asal sayı değerleri



Şekil 7.53. Civil Violence modelinde komşuluk alanı büyüklüğü 2 iken ve tampon bölgede birbirine komşu etmenler varken bir etkileşim maskesinin içerdiği asal sayı değerleri

İşlem gücü açısından bakıldığında ÖGS mekanizmasını kullanan bir Mİ aşağıdaki işlemlerden kaynaklı olarak işlemciye ek bir yük getirebilir.

- Etkileşim maskelerini oluşturmak için ortaya çıkan ek iş yükü: Etkileşim maskesine bir girdi yapılmasına neden olacak etmenler tampon alanda bulunan etmenlerdir. Tampon alandaki etmenler, komşu Mİ'nin bu etmenleri de hesaplamasında kullanabilmesi için zaten komşu Mİ'ye bildirilmektedir. Dolayısıyla bu etmenler komşu Mİ'ye gönderilmeden önce etkileşim maskesinin oluşturulmasında da kullanılırlar. Bu yüzden bu etmenlerin tespit edilmesi için herhangi bir ek iş yükü gerekmemektedir. Bu etmenler için gerekli asal sayıların bir veri yapısına (etkileşim maskesi) yerleştirilmesi ise ihmal edilebilecek kadar küçük bir iş yükü ortaya çıkarmaktadır.
- Maskeleme işlemi için ortaya çıkan ek iş yükü: Maskeleme işleminin yapılması için öncelikle bir mızıkçı ileti izi oluşturulması gerekmektedir. Mızıkçı iletinin içeriğindeki etmenlerin konumuna göre mızıkçı ileti izi içerisine bazı asal sayılar yerleştirilir. Bu işlem ihmal edilebilecek kadar küçük bir iş yüküne sahiptir. Ardından maskeleme işlemi için etkileşim maskesinin mızıkçı ileti izine bölünmesi gerekmektedir. Gerçekleştirilecek bölme işlemi sayısı ise en çok mızıkçı ileti izindeki değer sayısı kadar olabilir (örneğin; Şekil 7.3 – Şekil 7.32). Bir mızıkçı ileti, komşu Mİ'nin tampon alanının bir kopyasıdır; bu yüzden mızıkçı ileti izindeki ortalama değer sayısı tampon

alanda bulunan ortalama değer sayısına eşittir. Dolayısıyla herhangi bir t anında ortalama $A \cdot \frac{4b \cdot (N-b)}{N^2}$ adet değer mızıkçı ileti izinde bulunacağı için ortalama olarak bu sayıda bölme işlemi gerçekleştirileceği söylenebilir.

Yapılan bu incelemelere göre Civil Violence modeli ile yapılan örnek çalışmanın bellek ve işlem gücü açısından getirdiği maliyet hesaplanmıştır. Bellek kullanımı açısından ÖGS mekanizmasının Zaman Bükülmesi mekanizmasına göre ihmal edilebilir bir bellek alanı kaybı vardır. Örneğin; toplam etmen sayısının 129600, Mİ sayısının 16, uzay büyüklüğünün 3600×3600 ve komşuluk alanı büyüklüğünün 1 olduğu durum çalışmasında, bir Mİ t anında yaklaşık $A \cdot \frac{4b \cdot (N-b)}{N^2} \cdot b \cdot (2b+1) = 8100 \cdot \frac{4 \cdot (900-1)}{900^2} \cdot 1 \cdot (2+1) \cong 108$ adet değeri etkileşim maskelerinde tutmak zorundadır. Etkileşim maskesindeki değerler asal sayı olduğu için yazılımda birer işaretsiz tamsayı (İng. *unsigned integer*) türünde tutulmaktadır. Repast HPC yazılımını derlemede kullanılan gcc derleyicisinde işaretsiz tamsayılar 4 byte olarak kaydedilmektedir. Buna göre t anındaki etkileşim maskeleri için yaklaşık olarak 432 byte'lık bir alana ihtiyaç duyulduğu görülmektedir. Bu durum çalışmasında denetim noktası aralığı ise en çok 12 olarak ölçülmüştür. Dolayısıyla benzetim boyunca Zaman Bükülmesi mekanizmasına göre en çok 5184 byte'lık (≅5 KB) fazladan bir bellek alanına ihtiyaç duyulmuştur. Bu değer yapılan tüm durum çalışmaları arasında gözlenen en çok bellek ihtiyacının olduğu değerdir. Bir başka örnek olarak ise etmen sayısının 129600, Mİ sayısının 144, uzay büyüklüğünün 3600×3600 ve komşuluk alanı büyüklüğünün 1 olduğu durum çalışmasına bakılabilir. Bu örnekte ise bir Mİ'nin t anında etkileşim maskelerinde ortalama $A \cdot \frac{4b \cdot (N-b)}{N^2} \cdot b \cdot (2b+1) = 900 \cdot \frac{4 \cdot (300-1)}{300^2} \cdot 1 \cdot (2+1) \cong 36$ değer içerdiği bu değerlerin de 144 byte'ta tutulabileceği görülmüştür. Bu durum çalışmasında ulaşılan en yüksek denetim noktası aralığı 2 olarak ölçüldüğü için ise toplamda 288 byte'lık fazladan bellek alanı ihtiyacı olduğu ortaya çıkmaktadır. İşlem gücü açısından ÖGS mekanizmasının gerektirdiği ek yüke bakıldığında ise; etmen sayısının 129600, Mİ sayısının 16, uzay büyüklüğünün 3600×3600 ve komşuluk alanı büyüklüğünün 1 olduğu durum çalışmasında, t anında yapılması gereken en çok bölme işlemi sayısının $A \cdot \frac{4b \cdot (N-b)}{N^2} = 8100 \cdot \frac{4 \cdot (900-1)}{900^2} \cong 36$ kadar olduğu görülmektedir. Bu değer yapılan tüm

durum çalışmaları arasında en maliyetli olan deneye aittir. Öte yandan etmen sayısının 129600, Mİ sayısının 144, uzay büyüklüğünün 3600×3600 ve komşuluk alanı büyüklüğünün 1 olduğu durum çalışmasında ise $A \cdot \frac{4b \cdot (N-b)}{N^2} = 810 \cdot \frac{4 \cdot (900-1)}{900^2} \cong 12$ adet fazladan bölme işlemi maliyetinin ortaya çıktığı görülmektedir.

Günümüzdeki YBH sistemleri düşünüldüğünde hem bellek alanı açısından hem de işlem gücü açısından ortaya çıkan maliyetlerin kabul edilebilir değerler olduğu görülmektedir. Zaten Şekil 7.33 – Şekil 7.36'da verilen grafiklerde görüldüğü gibi, ÖGS mekanizmasının Zaman Bükülmesi mekanizmasına göre daha başarılı sonuçlar üretmesi de bu ek iş yüklerinin ÖGS için önemli bir olumsuzluk getirmediğinin bir göstergesidir.

7.6. Öngörülmüş Geri Sarma Mekanizmasının Farklı Tür Modellerde Kullanılmasına Yönelik Öneriler

Civil Violence örneği ÖGS mekanizmasının başarımını ölçmek için örnek olarak kullanılmış bir modeldir. Bu modelin, aşağıda verilen şu nedenlerden dolayı tercih edilmiştir:

- Model ayrıntılarıyla uğraşmak yerine ÖGS mekanizmasının etkilerinin görülebilmesine odaklanmak için Civil Violence modelinin gerçekleştiriminin ve etkileşim modelini tanımlamanın kolay olması,
- [120] kaynağında verilen çalışmada tanımlanması nedeniyle olgun bir model olarak değerlendirilmesi,
- Modelin farklı parametrelerini kullanarak zaman yönetim mekanizmasının farklı koşullar altındaki başarımını gözleme olanağı sağlaması.

ÖGS mekanizması sadece Civil Violence için değil, uzamsal-koşut olarak tanımlanabilecek bir çok etmen tabanlı model için de kullanılabilir durumdadır. ÖGS mekanizmasının farklı modeller için uygunluğunu gösterebilmek adına, örnek modeller sosyal benzetim modelleri, savunma uygulamaları ve oyunlar, moleküler benzetim modelleri ve hücresel özdevinir kullanan modeller olarak dört sınıf altında aşağıda incelenmiştir.

7.6.1. Sosyal Benzetim Modelleri

Etmen tabanlı modelleme ve benzetim sosyoloji, biyoloji, ulaştırma, iklim-çevre, finans vb. bir çok alandaki araştırmacılar tarafından yoğun olarak kullanılmaktadır. Bu modeller benzer özellikler taşıdıkları için sosyal benzetim modelleri adı altında bu alt başlıkta bir arada incelenmiştir.

Sosyal modeller incelendiğinde, bir çok modelde etmenlerin bir uzay üzerine yerleştirildiği ve komşuluk alanı üzerinden etkileşime girerek modellendiği görülmektedir. Dolayısıyla bu modellerin etkin bir şekilde koşutlaştırılması için uzamsal koşut yöntemlerden yararlanılması gerektiği değerlendirilmektedir. Bu modellerin bazı örnekleri şu şekilde verilebilir:

- Etmenler arası işbirliği ve rekabetin olduğu modeller (örneğin [133]),
- Etmenlerin belli kaynakları keşfetme amacında olduğu modeller (örneğin; yiyecek, maden, su, petrol keşfetme modelleri [109]),
- Yayılım modelleri (örneğin; bilgi yayılım modelleri, salgın hastalık yayılım modelleri [19][20]),
- Etmen davranışlarının ele alındığı sosyolojik modeller (örneğin; ırkçılık modelleri [130], seçim modelleri [109]),
- Trafik ve ulaşım modelleri (örneğin [22]),
- Av-avcı modelleri (örneğin; kurt-kuzu modeli [109]),
- Sürü davranışının benzetiminin yapıldığı modeller (örneğin; lideri takip etme modeli, hayvan sürülerinin hareketlerinin modeli [109]),
- Çevresel, ekolojik, biyolojik modeller (örneğin; orman yangını modelleri [129], hava durumu modelleri, popülasyon değişimini inceleyen modeller [109])
- İnsan davranışı modelleri (örneğin; kalabalık davranışı modelleri, acil durumda tahliye modelleri [134]).

Yukarıda verilen bu modellere ait basit gerçekleştirim örnekleri [109] kaynağında bulunabilir. Bu tarz modeller için etkileşim modellerinin tanımlanarak birçok gereksiz geri sarmadan kaçınmanın mümkün olduğu; dolayısıyla ÖGS mekanizması kullanarak uzamsal koşut çalışabilen bir çok sosyolojik modelin hızlandırılabileceği değerlendirilmektedir.

7.6.2. Savunma Uygulamaları ve Oyunlar

Savunma uygulamalarında fiziksel test süreçlerinden önce benzetimlere çok sık başvurulmakta olup, benzetimler bu süreçlerde en sağlıklı yöntemlerden biri olarak değerlendirilmektedir. Bu benzetimler, mühendislik benzetimleri (örneğin; uçuş yörüngelerinin hesaplandığı benzetimler, angajman modellerinin kullanıldığı benzetimler, harp oyunları) veya simülatörler için geliştirilen benzetim örnekleri olabilir. Bu tarz örneklerin koşut çalıştırılması değerlendirildiğinde, angajman modellerinin ve harp oyunlarının uzamsal koşut olarak ayrıştırılması mümkün gözükmektedir [135][136]. Bu modellerde etmenler arasındaki etkileşim modelinin de tanımlanarak ÖGS mekanizmasının oldukça verimli bir şekilde kullanılabileceği değerlendirilmektedir. Benzer durum son yıllarda sık kullanılan 3 boyutlu çevrimiçi oyunlar için de geçerlidir.

7.6.3. Moleküler Benzetim Modelleri

Moleküler benzetimler, özellikle akışkanlar dinamiğinin modellenmesinde yoğun olarak kullanılan benzetim örneklerinden birisidir. Akışkanlar dinamiği ile ilgilenen modellerde örgü (İng. *mesh*) tabanlı yöntemler veya parçacık tabanlı modeller kullanılabilmektedir. Parçacık tabanlı modeller, etmen tabanlı modellerle büyük oranda benzerlik göstermektedir. Örneğin Smoothed Particle Hydrodynamics [137] adı verilen yöntem ile parçacıklar kümesi olarak tanımlanan akışkanların modellemesi yapılabilmektedir. Parçacık tabanlı modeller, parçacıkların komşuluk üzerinden etkileşime girmesi dolayısıyla uzamsal-koşut çalıştırmaya da oldukça uygun modellerdir. [138], [139] ve [140] kaynaklarında uzamsal olarak koşutlaştırmaya yönelik örnekler bulunabilir. Parçacık tabanlı modellerde moleküller arasındaki etkileşim oldukça yoğun olabilir. Dolayısıyla etkileşimin yoğun olduğu parçacık tabanlı modellerde iyimser zaman yönetim mekanizmasının verimli olmayacağı değerlendirilmektedir. Çünkü, hem yoğun etkileşim Mİ'ler arasındaki iletişimin artmasına neden olacak; hem de geri sarma işlemleri, tüm moleküllerin benzetim durumunun geriye çekilerek benzetim adımlarının yeniden hesaplanmasına neden olacağı için çok pahalı olacaktır. Ayrıca parçacık tabanlı yöntemlerde tüm moleküller birbiriyle etkileşim içerisinde bulunacağından dolayı ÖGS mekanizması için gerekli olan etkileşim modelini tanımlamak mümkün olmayacaktır ve ÖGS mekanizması yalnızca uzamsal konumlara bakarak çalışacaktır. Bu nedenle ÖGS mekanizmasının parçacık yoğunluğunun yüksek

olduğu benzetimlere bir getiri sağlamayacağı değerlendirilmektedir. Ancak parçacık yoğunluğunun daha az olduğu ve parçacıkların Mİ'lere etkin bir şekilde dağıtıldığı modellerde iyimser zaman yönetim mekanizması tercih edilebilir olacaktır. Bu modellerde ÖGS mekanizmasının da kullanımıyla benzetimlerin sadece konumsal bilgileri kullanarak kaçınacağı geri sarma işlemleriyle bile benzetim hızını artırabileceği değerlendirilmektedir. Çünkü parçacık tabanlı modellerde geri sarma işlemlerinin oldukça pahalı olacağı ve bunlardan kaçınmanın da önemli bir getiri sağlayacağı değerlendirilmektedir. Özetle, parçacık tabanlı modellerde zaman yönetim mekanizması kullanılmadan önce özenli bir analiz çalışması yapmak gerektiği değerlendirilmektedir.

Akışkanlar dinamiğinin yanı sıra, farklı özelliklere sahip parçacıkların bulunduğu ve bu parçacıklar arasındaki etkileşimin parçacık niteliklerine bağlı olarak ele alındığı moleküler benzetim örnekleri de mevcuttur. Şekil 7.11 – Şekil 7.24'te verilmiş olan ve parçacıklar arasındaki kimyasal tepkimeler sonucu oluşacak denge durumunu gözlemlemeyi amaçlayan model bunun bir örneğidir. Bu tarz modeller anlamsallık bakımından sosyal benzetim modelleriyle çok benzerlik göstermektedir. Dolayısıyla, etkileşim modelinin tanımlanarak, ÖGS mekanizmasının bu tarz problemlerde oldukça başarılı olacağı değerlendirilmektedir.

7.6.4. Hücresel Özdevinir Kullanan Modeller

Hücresel özdevinirler uzun yıllardan bu yana çalışılmakta olup etmen tabanlı modelleme ile çok büyük benzerlikler göstermektedir. Conway'in Game of Life [141] adıyla geliştirdiği ve çok bilinen basit örneklerin yanısıra, hava durumu gibi gelişmiş modellerin de hücresel özdevinirler kullanarak geliştirildiği örneklere rastlamak mümkündür [142][143]. Hücresel özdevinirler genellikle uzaydaki tüm hücrelerin etrafındaki hücrelerle etkileşime girdiği modellerdir. Bu modelleri iyimser zaman yönetim mekanizmasıyla birlikte kullanan çalışmalara da rastlamak mümkündür [144]. Ancak, Mİ'ler tampon alanlarındaki etmenler dolayısıyla sürekli olarak komşularıyla etkileşime girmek durumundadırlar. Bölüm 7.5'te ortaya çıkan sonuçlar etkileşim olasılığının yüksek olduğu durumlarda ÖGS mekanizmasının Zaman Bükülmesi mekanizmasına göre kayda değer bir iyileştirme sağlamadığını göstermektedir. Dolayısıyla hücresel özdevinir modellerinde hücreler arası

etkileşim, buna bağlı olarak da Mİ'ler arasındaki etkileşim yoğun olacağı için ÖGS mekanizmasının kullanılmasının verimli olmayacağı değerlendirilmektedir.

8. SONUÇLAR VE TARTIŞMA

Bu tez çalışmasında YBH sistemleri için koşut ve dağıtılmış bir benzetim altyapısı geliştirilmiş ve benzetimlerin daha hızlı koşturulabilmesi için uzamsal-koşut modellere yönelik özgün yöntemler önerilmiştir. Önerilen yöntemler koşut ve dağıtılmış bir etmen tabanlı modelleme ve benzetim aracı olan açık kaynak kodlu Repast HPC yazılımı üzerine inşa edilmiştir.

Tez çalışması kapsamında sağlanan ilk katkı halihazırda sakıngan zaman yönetim mekanizmasına sahip Repast HPC yazılımına iyimser zaman yönetim desteği sağlanması olmuştur. Bunun için en bilinen iyimser zaman yaklaşımı olan Zaman Bükülmesi mekanizması Repast HPC'ye eklenmiştir. Bu gerçekleştirim, geliştiriciden soyut bir şekilde kullanılabilir durumdadır. Yani, geliştiriciler iyimser zaman yönetim mekanizmasının ayrıntılarını (geri sarma, denetim noktası alma, durum kaydetme, vb. işlemler) bilmek zorunda değildir. Böylece geliştiriciler benzetimi çalıştırmadan önce hangi zaman yönetim mekanizmasını kullanmak istediğini belirterek benzetimlerini koşturabilecek durumdadırlar. Zaman yönetim mekanizmalarının etmen tabanlı modeller üzerindeki başarımını görmek üzere yapılan deneylerde iyimser yaklaşımın ölçeklenebilir sonuçlar verdiği ve sakıngan yaklaşıma göre çok daha hızlı çalışabildiği durumların olduğu gözlenmiştir. Repast HPC'ye eklenen iyimser zaman yönetim mekanizması yalnızca uzamsal-koşut benzetimleri hedef almış, etmen-koşut benzetimler için herhangi bir geliştirme yapılmamıştır. Bu haliyle Repast HPC'nin daha gelişmiş mekanizmaları (geç iptal etme, fosil toplama, vb.) da içerecek şekilde genişletilmesi de mümkündür.

İyimser yaklaşımların başarımı açısından oldukça kritik olan noktalardan bir tanesi de denetim noktası alma mekanizmalarının sıklığıdır. Bilgisayar ağları teknolojilerinde iyi bilinen bir yöntem olan TAÇA algoritması denetim noktası alma sıklığının belirlenmesi için iyimser zaman yönetim mekanizmasında kullanılmıştır. TAÇA algoritması son zamanlarda gerçekleşen geri sarma işlemine göre denetim noktasının sıklığını artırmakta veya azaltmaktadır. Bu yöntemin en önemli avantajlarından bir tanesi, literatürdeki modellere göre geliştiriminin çok daha kolay olmasıdır. Ayrıca uzamsal-koşut modellerde denetim noktası sıklığının son

zamanlardaki geri sarma işlemlerine göre daha avantajlı olacağı değerlendirilmiştir. Çünkü, uzamsal-koşut modellerde Mİ'ler arasındaki etkileşim, Mİ'lerin yönettiği ortamların sınırlarına yakın alanlarda etmen bulunmasından kaynaklanır. Bu etkileşimlerin zamanında gerçekleşememesi geri sarma işlemlerine neden olmaktadır. Dolayısıyla bir geri sarma işleminin gerçekleşmesi; sınırlara yakın alanlarda etmenlerin bulunduğunu ve bu yüzden kısa vadede Mİ'ler arasında daha fazla etkileşim gerçekleşebileceğinin bir göstergesidir. Böyle bir durumda denetim noktası aralıklarının kısılarak Mİ'lerin fazla iyimser olması engellenir. Gelecek çalışmalar kapsamında TAÇA mekanizmasının, denetim noktası sıklığını sabit veya dinamik olarak belirleyen literatürdeki diğer mekanizmalarla başarım açısından daha ayrıntılı olarak karşılaştırılması planlanmaktadır.

Bu tez kapsamında hedeflenen ana katkı Zaman Bükülmesi mekanizmasının başarımını artırmak olarak belirlenmiştir. Uzamsal-koşut benzetimlerde geleneksel Zaman Bükülmesi mekanizmasının gereksiz yere bazı geri sarma işlemleri yapması sorunu ele alınmıştır. Bu nedenle oldukça maliyetli olabilen geri sarma işlemlerinin gerçekten gerekli olup olmadığının tespit edilmesi sağlanmıştır. Gereksiz geri sarma işlemlerini önleyerek benzetimlerin çalışma süresini kısaltmayı amaçlayan iki yöntem önerilmiştir:

- Önerilen yöntemlerden ilki bir durum kaydetme yöntemi olarak geliştirilmiş olup alt-durum kaydetme mekanizması olarak adlandırılmıştır. Alt-durum kaydetme mekanizması, Mİ'lerin tüm benzetim durumu yerine sadece belli etmenlerin durumunu kaydetmesini sağlar. Kaydedilen bu etmenler gelecekte alınacak mızıkçı iletilerin içereceği etmenlerle etkileşime girebilecek olan etmenlerdir. Alt-durum kaydetme mekanizması geleneksel Zaman Bükülmesi mekanizması ile karşılaştırıldığında ciddi bir hızlanma sağlayamadığı, ancak Mİ sayısı büyüdükçe hemen hemen aynı hızlanmaya ulaştığı görülmüştür.
- Önerilen ikinci yöntem ise Öngörülmüş Geri Sarma (ÖGS) mekanizması olarak adlandırılmıştır. Bu yöntem de alt-durum kaydetme mekanizmasına benzer şekilde ileride gelebilecek mızıkçı iletilerin hangi durumda geri sarma işlemine neden olacağını tespit etmeye çalışır. Böylece ileride gerçekleşebilecek geri sarmaların gerekliliğinin tespit edilebilmesi için vakitlice önlem alınmış olur. Yapılan deneyler sonucunda ÖGS

mekanizmasının geleneksel Zaman Bükülmesi mekanizmasına göre önemli oranda daha hızlı çalıştığı gözlenmiştir. Ayrıca yapılan hiçbir deneyde geleneksel Zaman Bükülmesi mekanizmasının ÖGS mekanizmasına göre kayda değer bir şekilde hızlı çalıştığı gözlenmemiştir. Özellikle Mİ'ler arasındaki etkileşim yoğunluğunun düşük olduğu durumlarda ÖGS mekanizmasının farkı iyice açtığı görülmüştür. Gözlemlenen bir diğer önemli nokta ise ÖGS mekanizmasının Zaman Bükülmesi mekanizmasına göre çok daha iyimser davranabilmesi olmuştur. Çünkü Mİ'lerin gereksiz geri sarma işlemlerinden kurtulması, TAÇA algoritmasının bir sonraki denetim noktasını daha uzak bir zamanda almasını sağlamaktadır. Bir başka deyişle, bir Mİ kısa vadede bir başka Mİ ile etkileşime girme olasılığının düşük olduğunu fark ederek daha iyimser davranmaktan çekinmemektedir. Bu iyimser davranmanın sonucunda da Mİ'ler koşut çalışmanın avantajından daha çok yararlanmakta ve benzetimleri daha kısa sürede tamamlayabilmektedir. Benzer şekilde Mİ'ler arasında etkileşim olasılığı yükseldiğinde, denetim noktası sıklığı artırılarak Mİ'lerin iyimserliği azaltılır. Böylece Mİ'ler biraz daha sakıngan davranarak maliyetli geri sarma işlemlerinin ortaya çıkmasına müsaade etmez.

Önerilen her iki yöntem de modelin anlamsallığını (İng. *semantics*) değiştirmeyecek şekilde geliştirilmiştir. Böylece benzetimlerin doğruluğundan ve tekrarlanabildiğinden taviz verilmeden bir iyileştirme sağlanmıştır.

ÖGS mekanizmasının gerektirdiği ek işlem gücü ve ek bellek yükünün analizi yapıldığında da oldukça başarılı olduğu değerlendirilmiştir. Hızlanma sonuçlarına bakıldığında, zaten geleneksel Zaman Bükülmesi mekanizmasına göre ÖGS mekanizmasının daha hızlı çalıştığı görülmektedir. ÖGS mekanizmasının gerektirdiği ek bellek yükü incelendiğinde ise günümüz YBH sistemleri için ihmal edilebilir bir seviyede maliyet getirdiği görülmüştür. Tüm bunlar göz önünde bulundurulduğunda uzamsal-koşut çalışan etmen tabanlı benzetimlerin ÖGS mekanizması kullanıldığında mevcut yöntemlerden daha başarılı sonuçlar verebildiği görülmüştür. Özellikle etkileşim yoğunluğunun düşük olduğu modellerde, mevcut yöntemlere oranla çok daha başarılı sonuçlar, ihmal edilebilecek seviyede ek bir yükle elde edilmiştir.

Mevcut haliyle ÖGS mekanizması etkileşim modelini dikkate almadan, yalnızca etmenlerin bulunduğu konuma göre de çalışabilmektedir. Bu yönüyle bakıldığında, ÖGS mekanizmasının geliştiricilerden tamamen soyutlandığı söylenebilir. Ancak etkileşim modeli sağlandığında daha fazla sayıda gereksiz geri sarma işlemi tespit edilebilmektedir. Dolayısıyla model geliştiriciler, benzetimlerini daha hızlı çalıştırmak isterlerse etkileşim modelini tanımlayabilirler. Bu yönüyle ÖGS mekanizması geliştiriciden yüzde yüz soyutlanmış değildir. Ancak etkileşim modelinin belirlenmesi için yapılması gereken şey, sadece hangi etmenler arasında etkileşim olacağını tanımlayan işlevin benzetim modeline tanımlanmasıdır. Yine de geliştiricilerin işini kolaylaştırabilmek için bir modelleme dili geliştirilerek etkileşim modelinin bu dil aracılığıyla tanımlanması sağlanabilir. Sonrasında da bu etkileşim modelinin C++ koduna otomatik olarak dönüştürülmesi ile geliştiricinin zaman yönetiminden bir kat daha soyutlanması sağlanabilir. Etkileşim modeli tanımlamak için böyle bir arayüzün geliştirilmesi gelecek çalışma planları arasına alınmıştır.

KAYNAKLAR

[1] R. M. Fujimoto, Parallel and Distributed Simulation Systems, 1. Baskı, John Wiley and Sons, Inc., **2000**.

[2] K.S. Perumalla, Scaling Time Warp-Based Discrete Event Execution to 104 Processors on a Blue Gene Supercomputer, Proceedings of the 4th International Conference on Computing Frontiers, Ischia, Italy, 69-76, **2007**.

[3] P.D. Barnes Jr., C.D. Carothers, D.R. Jefferson, J.M. LaPre, Warp Speed: Executing Time Warp on 1,966,080 Cores, Proceedings of the 1st ACM SIGSIM Conference on Principles of Advanced Discrete Simulation, Montreal, Quebec, Canada, 327-336, **2013**.

[4] M. H. Kalos, P. A. Whitlock. 2008. Monte Carlo Methods, Weinheim, John Wiley and Sons, Inc., 2. Baskı, **2008**.

[5] IEEE 1516-2010, IEEE Standard for Modelling and Simulation (M&S) High Level Architecture (HLA)--Framework and Rules, **2010**.

[6] IEEE 1278-1993 - IEEE Standard for Information Technology - Protocols for Distributed Interactive Simulations Applications. Entity Information and Interaction, **1993**.

[7] D.A. Greschke, S. Cerutti, Aircrew Mission Training via Distributed Simulation (MTDS)- Development of the Multi-Country Complex Synthetic Environment, RTO HFM Symposium on Advanced Technologies for Military Training, 13-15 Ekim, Genoa, Italy, 18:1–18:18, **2003**.

[8] B. Tomlinson, J. van Geest, Aircrew Mission Training via Distributed Simulation Progress in NATO, RTO HFM Symposium on Advanced Technologies for Military Training, 13-15 Ekim, Genoa, Italy, 16:1–16:14, **2003**.

[9] T. Ören, L. Yilmaz, Synergies of Simulation, Agents, and Systems Engineering, Expert Systems with Applications, 39(**2012**), 1, 81-88.

[10] C.M. Macal, M.J. North, Agent-Based Modeling and Simulation, Proceedings of Winter Simulation Conference (WSC'09), 13-16 Aralık, Austin, Texas, Amerika Birleşik Devletleri, 86-98, **2009**.

[11] C.M. Macal, M.J. North, Tutorial on agent-based modeling and simulation, Journal of Simulation 4(**2010**), 3, 151-162.

[12] N.A. Fates, A guided tour of asynchronous cellular automata, Cellular Automata and Discrete Complex Systems (**2013**), 15-30.

[13] A. Rousset, B. Herrmann, C. Lang, L. Philippe, A survey on parallel and distributed multi-agent systems for high performance computing simulations, Computer Science Review 22(**2016**), 27-46.

[14] H.R. Parry, M. Bithell, in Agent-Based Models of Geographical Systems, A. Heppenstall, A. Crooks, L. See, M. Batty (Eds.), Chapter: Large Scale Agent-Based Modelling: A Review and Guidelines for Model Scaling 271-308, **2012**.

[15] N. Fachada, V.V. Lopes, R.C. Martins, A.C. Rosa, Parallelization Strategies for Spatial Agent-Based Models, International Journal of Parallel Programming, 45(**2017**), 3, 449-481.

[16] G. Chen, M.A. Anders, H. Kaul, S.K. Satpathy, S.K. Mathew, S.K. Hsu, A. Agarwal, R.K. Krishnamurthy, V. De, S. Borkar, A 340 mV-to-0.9 V 20.2 Tb/s Source-Synchronous Hybrid Packet/Circuit-Switched 16 × 16 Network-on-Chip in 22 nm Tri-Gate CMOS, IEEE Journal of Solid-State Circuits, 50(**2015**), 1, 59-67.

[17] B. Bohnenstiehl, A. Stillmaker, J. Pimentel, T. Andreas, B. Liu, A. Tran, E. Adeagbo, B. Baas, KiloCore: A Fine-Grained 1,000 Processor Array for Task-Parallel Applications, IEEE Micro, 37(**2017**), 2, 63-69.

[18] B.J. Overeinder, P.M.A. Sloot, Application of Time Warp to Parallel Simulations with Asynchronous Cellular Automata, Proceedings of the 1993 European Simulation Symposium, 397-402, **1993**.

[19] D.M. Rao, Accelerating Parallel Agent-based Epidemiological Simulations, Proceedings of the 2nd ACM SIGSIM Conference on Principles of Advanced Discrete Simulation, Denver, Colorado, Amerika Birleşik Devletleri, 127-138, **2014**.

[20] D.M. Rao, Efficient parallel simulation of spatially-explicit agent-based epidemiological models, Journal of Parallel and Distributed Computing, 93-94(**2016**), 102-119.

[21] T.W. Clark, R. von Hanxleden, J.A. McCammon, L.R. Scott, Parallelizing molecular dynamics using spatial decomposition, Proceedings of IEEE Scalable High Performance Computing Conference, 23-25 Mayıs, Knoxville, TN, Amerika Birleşik Devletleri, 95-102, **1994**.

[22] Y. Xu, W. Cai, H. Aydt, M. Lees, D. Zehe, Relaxing Synchronization in Parallel Agent-Based Road Traffic Simulation, ACM Transactions on Modeling and Computer Simulation (TOMACS), 27(**2017**), 2, 14:1-14:24.

[23] P. Bauer, J. Lindèn, S. Engblom, B. Jonsson, Efficient Inter-Process Synchronization for Parallel Discrete Event Simulation on Multicores, Proceedings of the 3rd ACM SIGSIM Conference on Principles of Advanced Discrete Simulation, 183-194, **2015**.

[24] J. Lindèn, P. Bauer, S. Engblom, B. Jonsson, Exposing Inter-Process Information for Efficient Parallel Discrete Event Simulation of Spatial Stochastic Systems, Proceedings of the 2017 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation, 53-64, **2017**.

[25] P. Andelfinger, Y. Xu, W. Cai, D. Eckhoff, A. Knoll, Fast-Forwarding Agent States to Accelerate Microscopic Traffic Simulations, Proceedings of the 2018 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation, 23-25 Mayıs, Roma, Italy, 113-124, **2018**.

[26] N. Collier, M. North, Parallel agent-based simulation with Repast for High Performance Computing, SIMULATION, 89(**2013**), 10, 1215-1235.

[27] D.M. Chiu, R. Jain, Analysis of the increase and decrease algorithms for congestion avoidance in computer networks, Computer Networks and ISDN Systems, 17(**1989**), 1, 1-14.

[28] S. Jafer, Q. Liu, G. Wainer, Synchronization Methods in Parallel and Distributed Discrete-Event Simulation, Simulation Modelling Practice and Theory, 30(**2013**), 54-73.

[29] A. Tolk, Engineering Principles of Combat Modeling and Distributed Simulation, John Wiley & Sons, Inc., 1. Baskı, **2012**.

[30] R.M. Fujimoto, Performance Measurements of Distributed Simulation Strategies, Technical Report, Utah University Salt Lake City Department of Computer Scıence, Salt Lake City, Utah, USA, **1987**.

[31] D.R. Jefferson, Virtual Time. ACM Transactions on Programming Languages and Systems (TOPLAS), 7(**1985**), 3, 404-425.

[32] R.M. Fujimoto, Parallel and Discrete Event Simulation, Communications of the ACM, 33(**1990**), 10, 30-53.

[33] D. West, Optimising Time Warp: Lazy Rollback and Lazy Reevaluation, Yüksek Lisans Tezi, The University of Calgary Faculty of Graduate Studies, Calgary, Canada, **1988**.

[34] R.M. Fujimoto, Distributed Simulation Systems, Proceedings of the 35th Conference on Winter Simulation: Driving Innovation, New Orleans, Louisiana, 124-134, **2003**.

[35] C.D. Carothers, K.S. Perumalla, On Deciding Between Conservative and Optimistic Approaches on Massively Parallel Platforms, Proceedings of the 2010 Winter Simulation Conference, 5-8 Aralık, Baltimore, Maryland, USA, 678-687, **2010**.

[36] S. Jafer, G. Wainer, Conservative vs. Optimistic Parallel Simulation of DEVS and Cell-DEVS: A Comparative Study, Proceedings of the 2010 Summer Computer Simulation Conference, 11-14 Temmuz, Ottawa, Ontario, Canada, 342-349, **2010**.

[37] M.T. Presley, P.L. Reiher, S.F. Bellenot, A Time Warp Implementation of Sharks World. 1990 Winter Simulation Conference Proceedings, 9-12 Aralık, New Orleans, Los Angeles, USA, 199-203, **1990**.

[38] D.W. Bauer Jr., C.D. Carothers, A. Holder, Scalable Time Warp on Blue Gene Supercomputers, 2009 ACM/IEEE/SCS 23rd Workshop on Principles of Advanced and Distributed Simulation, 22-25 Haziran, Lake Placid, New York, USA, 35-44, **2009**.

[39] H. Avril, C. Tropper, On Rolling Back and Checkpointing in Time Warp, IEEE Transactions on Parallel and Distributed Systems, 12(**2001**), 11, 1105-1121.

[40] E.N. Elnozahy, L. Alvisi, Y-M. Wang, D.B. Johnson, A Survey of Rollback-Recovery Protocols in Message-Passing Systems, ACM Computing Surveys, 34(**2002**), 3, 375-408.

[41] B.R. Preiss, W.M. Loucks, I.D. MacIntyre, Effects of the Checkpoint İnterval on Time and Space in Time Warp, ACM Transactions on Modeling and Computer Simulation (TOMACS), 4(**1994**), 3, 223-253.

[42] R. Rönngren, R. Ayani, Adaptive Checkpointing in Time Warp, ACM SIGSIM Simulation Digest, 24(**1994**), 1, 110-117.

[43] J. Fleischmann, P.A. Wilsey, Comparative Analysis of Periodic State Saving Techniques in Time Warp Simulators, Proceedings of the 9th Workshop on Parallel and Distributed Simulation, 14-16 Haziran, Lake Placid, New York, USA, 50-58, **1995**.

[44] B.R. Preiss, I.D. MacIntyre, W.M. Loucks, On the Trade-off between Time and Space in Optimistic Parallel Discrete-Event Simulation, 6th Workshop on Parallel and Distributed Simulation, 33-42, **1992**.

[45] Y-B. Lin, B.R. Preiss, W.M. Loucks, E.D. Lazowska, Selecting the Checkpoint Interval in Time Warp Simulation, ACM SIGSIM Simulation Digest, 23(**1993**), 1, 3-10.

[46] F. Quaglia, A cost model for selecting checkpoint positions in time warp parallel simulation, IEEE Transactions on Parallel and Distributed Systems, 12(**2001**), 4, 346-362.

[47] L.R.G. Auriche, F. Quaglia, B.Ciciani, Run-time selection of the checkpoint interval in Time Warp based simulations, Simulation Practice and Theory, 6(**1998**), 5, 461-478.

[48] R. Suzuki, S. Fukurnoto, K. Iwasaki, Proceedings 22nd International Conference on Distributed Computing Systems Workshops, 2-5 Temmuz, Vienna, Austria, 95-100, **2002**.

[49] S.W. Kwak, J-M. Yang, Optimal Checkpoint Placement on Real-Time Tasks with Harmonic Periods, Journal of Computer Science and Technology, 27(**2012**), 1, 105-112.

[50] A. Ferscha, S.K. Tripathi, Parallel and Distributed Simulation of Discrete Event Systems, Technical Report, University of Maryland Institute for Advanced Computer Studies Report No. UMIACS-TR-94-100, College Park, Maryland, USA, **1994**.

[51] K.S. Perumalla, Parallel and Distributed Simulation: Traditional Techniques and Recent Advances, Proceedings of the 2006 Winter Simulation Conference, 3-6 Aralık, Monterey, California, USA, **2006**.

[52] J. Lindèn, Synchronization Techniques in Parallel Discrete Event Simulation, Doktora Tezi, Uppsala University, Department of Information Technology, Uppsala, Sweden, **2018**.

[53] R.M. Fujimoto, Parallel and Distributed Simulation, Proceedings of the 2015 Winter Simulation Conference, 6-9 Aralık, Huntington Beach, California, USA, 45-59, **2015**.

[54] M. Plagge, C.D. Carothers, E. Gonsiorowski, N. Mcglohon, NeMo: A Massively Parallel Discrete-Event Simulation Model for Neuromorphic Architectures, ACM Transactions on Modeling and Computer Simulation (TOMACS), 28(**2018**), 4, 30:1-30:25.

[55] R. Radhakrishnan, T.J. McBrayer, K. Subramani, M. Chetlur, V. Balakrishnan, P.A. Wilsey, A Comparative Analysis of Various Time Warp Algorithms Implemented in the Warped Simulation Kernel, Proceedings of the 29th Annual Simulation Symposium, 8-11 Aralık, New Orleans, Los Angeles, USA, 107-116, **2002**.

[56] J.S. Steinman, SPEEDES - A multiple-synchronization environment for parallel discrete-event simulation, International Journal in Computer Simulation, 2(**1992**).

[57] M. Damitio, S.J. Turner, Comparing the Breathing Time Buckets Algorithm and the Time Warp Operating System on a Transputer Architecture, Proceedings of the SCS European Simulation Multiconference, 1-3 Haziran, Barcelona, Spain, 141-145, **1994**.

[58] J.S. Steinman, Breathing Time Warp, ACM SIGSIM Simulation Digest, 23(**1993**), 1, 109-118.

[59] S. Srinivasan, P.F. Reynolds Jr., Adaptive algorithms vs. Time Warp: an analytical comparison, Proceedings of the 27th conference on Winter simulation, 3-6 Aralık, Arlington, Virginia, USA, 666-673, **1995**.

[60] P.F. Reynolds Jr., A spectrum of options for parallel simulation, Proceedings of the 20th conference on Winter simulation, 12-14 Aralık, San Diego, California, USA, 325-332, **1998**.

[61] S. Srinivasan, P.F. Reynolds Jr., Elastic Time, ACM Transactions on Modeling and Computer Simulation (TOMACS) - Special issue on modeling and analysis of stochastic systems, 8(**1998**), 2, 103-139.

[62] D. Jefferson, Virtual Time II: Storage Management in Conservative and Optimistic Systems, Proceedings of the Ninth Annual ACM Symposium on Principles of Distributed Computing, 75-89, **1990**.

[63] D. Jefferson, P.D. Barnes, Virtual time III: Unification of conservative and optimistic synchronization in parallel discrete event simulation, 2017 Winter Simulation Conference (WSC), 3-6 Aralık, Las Vegas, Nevada, USA, **2017**.

[64] A. Gafni, Rollback mechanisms for optimistic distributed simulation systems, Proceedings of the SCS Multiconference on Distributed Simulation, 61-67, **1988**.

[65] Y-B. Lin, E.D. Lazowska, A Study of Time Warp Rollback Mechanisms, ACM Transactions on Modeling and Computer Simulation (TOMACS), 1(**1991**), 1, 51-72.

[66] M.S. Balsamo, C. Manconi, Rollback overhead reduction methods for time warp distributed simulation, Simulation Practice and Theory, 6(**1998**), 8, 689-702.

[67] H.V. Leong, D. Agrawal, J.R. Agre, Using message semantics to reduce rollback in the time warp mechanism, Distributed Algorithms, A. Schiper (Eds), Springer, Berlin, Heidelberg, 309-323, **1993**.

[68] H.V. Leong, D. Agrawal, Using message semantics to reduce rollback in optimistic message logging recovery schemes, 14th International Conference on Distributed Computing Systems, 21-24 Haziran, Pozman, Poland, 227-234, **1994**.

[69] G. Chen, B.K. Szymanski, Lookback:a new way of exploiting parallelism in discrete event simulation, Proceedings of the 16th Workshop on Parallel and Distributed Simulation, 12-15 Mayıs, Washington, DC, USA, 138-147, **2002**.

[70] G. Chen, B.K. Szymanski, Four Types of Lookback, Proceedings of the 17th Workshop on Parallel and Distributed Simulation, San Diego, California, USA, 3-10, **2003**.

[71] R.M. Fujimoto, The Virtual Time Machine, Proceedings of the First Annual ACM Symposium on Parallel Algorithms and Architectures, 18-21 Haziran, Santa Fe, New Mexico, USA, 199-208, **1989**.

[72] D. Bruce, The Treatment of State in Optimistic Systems, SIGSIM Simulation Digest, 25(**1995**), 1, 40-49.

[73] H. Mehl, S. Hammes, How to Integrate Shared Variables in Distributed Simulation, SIGSIM Simulation Digest, 25(**1995**), 2, 14-41.

[74] N.V. Thondugulam, D.M. Rao, R. Radhakrishnan, P.A. Wilsey, Relaxing causal constraints in PDES, Proceedings 13th International Parallel Processing Symposium and 10th Symposium on Parallel and Distributed Processing. IPPS/SPDP 1999, 12-16 Nisan, San Juan, Puerto Rico, USA, 696-700, **1999**.

[75] F. Quaglia, R. Baldoni, Exploiting intra-object dependencies in parallel simulation, Information Processing Letters, 70(**1999**), 3, 119-125.

[76] R.M. Fujimoto, Exploiting Temporal Uncertainty in Parallel and Distributed Simulations, Proceedings of the Thirteenth Workshop on Parallel and Distributed Simulation, 1-4 Mayıs, Atlanta, Georgia, USA, 46-53, **1999**.

[77] M.L. Loper, R.M. Fujimoto, A case study in exploiting temporal uncertainty in parallel simulations, International Conference on Parallel Processing (ICPP 2004), 15-18 Ağustos, Montreal, Quebec, Canada, 161-168, **2004**.

[78] R. Beraldi, L. Nigro, Exploiting Temporal Uncertainty in Time Warp Simulations, Proceedings of the Fourth IEEE International Workshop on Distributed Simulation and Real-Time Applications (DS-RT 2000), 24-26 Ağustos, San Francisco, California, USA, 39-46, **2002**.

[79] R. Beraldi, L. Nigro, A. Orlando, F. Pupo, Temporal Uncertainty Time Warp: An Agent-Based Implementation, Proceedings of the 35th Annual Simulation Symposium, 14-18 Nisan, San Diego, California, USA, 72-79, **2002**.

[80] F. Cicirelli, A. Furfaro, L. Nigro, Distributed simulation of modular time Petri nets: An approach and a case study exploiting temporal uncertainty, Real-Time Systems, 35(**2007**), 2, 153-179.

[81] P. Pecher, M. Hunter, R. Fujimoto, Efficient Execution of Replicated Transportation Simulations with Uncertain Vehicle Trajectories, Procedia Computer Science, 51(**2015**), 2638-2647.

[82] F. Quaglia, R. Beraldi, Space uncertain simulation events: some concepts and an application to optimistic synchronization, 18th Workshop on Parallel and Distributed Simulation, 16-19 Mayıs, Kufstein, Austria, 181-188, **2004**.

[83] A. Pellegrini, Techniques for Transparent Parallelization of Discrete Event Simulation Models, Doktora Tezi, Sapienza University of Rome Faculty of Computer Engineering, Rome, Italy, **2014**.

[84] A. Pellegrini, F. Quaglia, Programmability and Performance of Parallel ECS-Based Simulation of Multi-agent Exploration Models, Euro-Par 2014: Parallel Processing Workshops, 25-26 Ağustos, 395-406, **2014**.

[85] A. Pellegrini, S. Peluso, F. Quaglia, R. Vitali, Transparent Speculative Parallelization of Discrete Event Simulation Applications Using Global Variables, International Journal of Parallel Programming, 44(**2016**), 6, 1200-1247.

[86] D. Cingolani, Application Transparent and Efficient Mixed State-saving in Speculative Simulation Platforms, Yüksek Lisans Tezi, Sapienza University of Rome Faculty of Computer Engineering, Rome, Italy, **2014**.

[87] R. Rönngren, M. Liljenstam, R. Ayani, J. Montagnat, Transparent Incremental State Saving in Time Warp Parallel Discrete Event Simulation, SIGSIM Simulation Digest, 26(**1996**), 1, 70-77.

[88] L. Chen, T. Lu, Y. Yao, S. Peng, L. Wu, A Well-Balanced Time Warp System on Multi-Core Environments, Proceedings of the 2011 IEEE Workshop on Principles of Advanced and Distributed Simulation, 14-17 Haziran, Nice, France, 1-9, **2011**.

[89] F. Nobilia, Runtime management of simulation objects' cross state dependencies in NUMA oriented parallel simulation platforms, Yüksek Lisans Tezi, Sapienza University of Rome Faculty of Computer Engineering, Rome, Italy, **2015**.

[90] M. Ianni, R. Marotta, D. Cingolani, A. Pellegrini, F. Quaglia, The Ultimate Share-Everything PDES System, Proceedings of the 2018 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation, 23-25 Mayıs, Rome, Italy, 73-84, **2018**.

[91] A. Pellegrini, F. Quaglia, A Fine-Grain Time-Sharing Time Warp System, ACM Transactions on Modeling and Computer Simulation (TOMACS) - Special Issue on PADS 2015, 27(**2017**), 2, 10:1-10:25.

[92] R. Child, P. Wilsey, Dynamically Adjusting Core Frequencies to Accelerate Time Warp Simulations in Many-Core Processors, Proceedings of the 2012 ACM/IEEE/SCS 26th Workshop on Principles of Advanced and Distributed Simulation, 15-19 Temmuz, Zhangjiajie, China, 35-43, **2012**.

[93] R.M. Fujimoto, Distributed Simulation Systems, Proceedings of the 2003 Winter Simulation Conference, 7-10 Aralık, New Orleans, Louisiana, USA, 124-134, **2003**.

[94] A. Malik, A. Park, R. Fujimoto, Optimistic Synchronization of Parallel Simulations in Cloud Computing Environments, 2009 IEEE International Conference on Cloud Computing, 21-25 Aralık, Bangalore, India, 49-56, **2009**.

[95] S.B. Yoginath, K.S. Perumalla, Empirical evaluation of conservative and optimistic discrete event execution on cloud and VM platforms, Proceedings of the 1st ACM SIGSIM Conference on Principles of Advanced Discrete Simulation, 19-22 Mayıs, Montreal, Quebec, Canada, 201-210, **2013**.

[96] H. Rajaei, Local Time Warp: An Implementation and Performance Analysis, 21st International Workshop on Principles of Advanced and Distributed Simulation (PADS'07), 12-15 Haziran, San Diego, California, USA, 163-170, **2007**.

[97] X. Liu, P. Andelfinger, Time Warp on the GPU: Design and Assessment, Proceedings of the 2017 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation, 24-26 Mayıs, Singapore, Republic of Singapore, 109-120, **2017**.

[98] D.M. Rao, Performance comparison of Cross Memory Attach capable MPI vs. Multithreaded Optimistic Parallel Simulations, Proceedings of the 2018 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation, 23-25 Mayıs, Rome, Italy, 37-48, **2018**.

[99] A. Helleboogh, T. Holvoet, D. Weyns, Y. Berbers, Extending Time Management Support for Multi-agent Systems, International Workshop on Multi-Agent Systems and Agent-Based Simulation, 37-48, **2004**.

[100] B. Logan, G. Theodoropoulos, The distributed simulation of multiagent systems, Proceedings of the IEEE, 89(**2001**), 2, 174-185.

[101] D. Pawlaszczyk, I.J. Timm, A Hybrid Time Management Approach to Agent-Based Simulation, KI 2006: Advances in Artificial Intelligence, Lecture Notes in Computer Science, C. Freksa, M. Kohlhase, K. Schill (Eds), Springer, Berlin, Heidelberg, 374-388, **2006**.

[102] D. Pawlaszczyk, S. Strassburger, Scalability in distributed simulations of agent-based models, Proceedings of the 2009 Winter Simulation Conference (WSC), 13-16 Aralık, Austin, Texas, USA, 1189-1200, **2010**.

[103] C. Sherer, G. Vulov, M. Hybinette, On-the-fly parallelization in agent-based simulation systems, Proceedings of the 2011 Winter Simulation Conference (WSC), 11-14 Aralık, Phoenix, Arizona, USA, 3017-3024, **2011**.

[104] M. Lees, B. Logan, R. Minson, T. Oguara, G. Theodoropoulos, Distributed Simulation of MAS, Proceedings of the 2004 International conference on Multi-Agent and Multi-Agent-Based Simulation, 25-36, **2004**.

[105] M. Lees, Adaptive optimistic simulation of multi-agent system, Doktora Tezi, The University of Nottingham, Nottingham, UK, **2006**.

[106] M. Lees, B. Logan, C. Dan, T. Oguara, G. Theodoropoulos, Analysing the Performance of Optimistic Synchronisation Algorithms in Simulations of Multi-Agent Systems, 20th Workshop on Principles of Advanced and Distributed Simulation (PADS'06), 24-26 Mayıs, Singapore, Singapore, 37-44, **2006**.

[107] M. Lees, B. Logan, G. Theodoropoulos, Adaptive optimistic synchronisation for multiagent distributed simulation, Proceedings of the 5th EUROSIM Congress on Modelling and Simulation (EuroSim'04), **2004**.

[108] T. Filatova, P.H. Verburg, D.C. Parker, C.A. Stannard, Spatial Agent-Based Models for Socio-Ecological Systems: Challenges and Prospects, Environmental Modelling & Software, 45(**2013**), 1-7.

[109] U. Wilensky, NetLogo, http://ccl.northwestern.edu/netlogo/ (Erişim tarihi: **22 Mart 2019**).

[110] George Mason University's Evolutionary Computation Laboratory & GMU Center for Social Complexity, MASON, https://cs.gmu.edu/~eclab/projects/mason/ (Erişim tarihi: **22 Mart 2019**).

[111] Argonne National Laboratory, Repast Simphony, https://repast.github.io/repast_simphony.html (Erişim tarihi: **22 Mart 2019**).

[112] ISIS Lab &The Department of Computer Science of University of Salerno & MASON, D-MASON, https://sites.google.com/site/distributedmason/ (Erişim tarihi: **22 Mart 2019**).

[113] Rensselaer Polytechnic Institute, Rensselaer's Optimistic Simulation System (ROSS), https://github.com/ROSS-org/ROSS, (Erişim tarihi: **22 Mart 2019**).

[114] Georgia Institute of Technology, Georgia Tech Time Warp, https://www.cc.gatech.edu/computing/pads/tech-parallel-gtw.html (Erişim tarihi: **22 Mart 2019**).

[115] M. Hybinette, E. Kraemer, Y. Xiong, G. Matthews, J. Ahmed, SASSY: a design for a scalable agent-based simulation system using a distributed discrete event infrastructure, Proceedings of the 38th conference on Winter simulation, 3-6 Aralık, Monterey, California, USA, 926-933, **2006**.

[116] G. Vulov, T. He, M. Hybinette, Quantitative assessment of an agent-based simulation on a time warp executive, 2008 Winter Simulation Conference, 7-10 Aralık, Miami, Florida, USA, 1068-1076, **2008**.

[117] P.A. Wilsey, WARPED, https://github.com/wilseypa/warped (Erişim tarihi: **22 Mart 2019**).

[118] A. Pellegrini, F. Quaglia, The ROme OpTimistic Simulator: A Tutorial, Proceedings of the European Conference on Parallel Processing, 501-512, **2013**.

[119] M.R. Gebre, MUSE: A parallel Agent-based Simulation Environment, Yüksek Lisans Tezi, Miami University Department of Computer Science and System Analysis, Oxford, Ohio, USA, **2009**.

[120] J.M. Epstein, Modeling civil violence: An agent-based computational approach, Proceedings of the National Academy of Sciences of the United States of America, 99(**2002**), suppl 3, 7243-7250.

[121] A. Grama, A. Gupta, G. Karypis, V. Kumar, Introduction to Parallel Computing, 2. Baskı, Pearson Education – Addison Wesley, **2003**.

[122] G.M. Amdahl, Validity of the single processor approach to achieving large scale computing capabilities, Proceedings of the April 18-20, 1967, Spring Joint Computer Conference, 18-20 Nisan, Atlantic City, New Jersey, USA, 483-485, **1967**.

[123] B. K. Görür, K. M. İmre, H. Oğuztüzün, L. Yılmaz, Repast HPC with Optimistic Time Management, Proceedings of the 24th High Performance Computing Symposium (SpringSim, HPC '16), 3-6 Nisan, San Diego, California, USA, 398-406, **2016**.

[124] T. Kiesling, S. Pohl, Time-parallel simulation with approximative state matching, Proceedings of the eighteenth workshop on Parallel and distributed simulation, 16-19 Mayıs, Kufstein, Austria, 195-202, **2004**.

[125] The GCC Project, GNU Compiler Collection (GCC) Internals 4.8.5, https://gcc.gnu.org/onlinedocs/gcc-4.8.5/gccint/ (Erişim tarihi: **22 Mart 2019**).

[126] SchedMD, Slurm Workload Manager, https://slurm.schedmd.com/ (Erişim tarihi: **22 Mart 2019**).

[127] B. K. Görür, K. M. İmre, H. Oğuztüzün, L. Yılmaz, Improving the Performance of Optimistic Time Management Mechanism with Sub-State Saving, Proceedings of the 25th High Performance Computing Symposium (SpringSim, HPC '17), 23-26 Nisan, Virginia Beach, Virginia, USA, 11:1-11:2, **2017**.

[128] U. Wilensky, NetLogo Chemical Equilibrium Model, http://ccl.northwestern.edu/netlogo/models/ChemicalEquilibrium (Erişim tarihi: **22 Mart 2019**).

[129] B. Drossel, F. Schwabl, Self-organized critical forest-fire model, Physical Review Letters, 69(**1992**), 11, 1629-1632.

[130] R.A. Hammond, R. Axelrod, The Evolution of Ethnocentrism, Journal of Conflict Resolution, 50(**2006**), 6, 926-936.

[131] S.S. Rizvi, A. Riasat, K.M. Elleithy, An efficient optimistic time management algorithm for discrete-event simulation system, International Journal of Simulation Modelling, 9(**2010**), 3, 117-130.

[132] B. K. Görür, K. M. İmre, H. Oğuztüzün, L. Yılmaz, Predetermined Rollbacks: An Extension to Time Warp for Spatially Parallel Agent-Based Simulation, Simulation Modelling Practice and Theory, 95(**2019**), 60-77, DOI: 10.1016/j.simpat.2019.04.008.

[133] J. Mitteldorf, D. S. Wilson, Population Viscosity and the Evolution of Altruism, Journal of Theoretical Biology, 204(**2000**), 4, 481-496.

[134] O. Szymanezyk, T. Duckett, P. Dickinson, Agent-Based Crowd Simulation in Airports Using Games Technology. Transactions on Computational Collective Intelligence VIII, N-T. Nguyen (Eds.), Vol. 7430, Springer, Berlin, Heidenberg, 192-213, **2012**.

[135] S. Zhou, W. Cai, S. J. Turner, J. Wei, W. Zong, Flexible State Update Mechanism for Large-Scale Distributed Wargame Simulations, Simulation, 83(**2007**), 10, 707-719.

[136] B. S. Onggo, M. Karataş, Test-driven simulation modelling: A case study using agent-based maritime search-operation simulation, European Journal of Operational Research, 254(**2016**), 2, 517-531.

[137] G. Liu, M. B. Liu, Smoothed Particle Hydrodynamics-A Meshfree Particle Method, World Scientific Publishing Co. Pte. Ltd., **2003**.

[138] M. Ihmsen, N. Akinci, M. Becker, M. Teschner, A Parallel SPH Implementation on Multi-Core CPUs, Computer Graphics Forum, 30(**2011**), 1, 99-112.

[139] G. Akinci, Efficient Surface Reconstruction for SPH Fluids, Doktora Tezi, University of Freiburg, Freiburg, Germany, **2014**.

[140] P. Goswami, P. Schlegel, B. Solenthaler, R. Pajarola, Interactive SPH Simulation and Rendering on the GPU, Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, 2-4 Temmuz, Madrid, Spain, 55-64, **2010**.

[141] J. Conway, The Game of Life, Scientific American, 223(**1970**), 4.

[142] H. Li, G. Corzo, C. M. Cano, A. Mynett, Self-Learning Cellular Automata for Forecasting Precipitation from Radar Images, Journal of Hydrologic Engineering, 18(**2013**), 2, 206-211.

[143] D. Robinson, Modelling Global Climate Variables with Cellular Automata Networks, Lisans Tezi, The University of Mississippi, Oxford, Mississippi, USA, **2015**.

[144] B. J. Overeinder, P. M. A. Sloot, Application Of Time Warp To Parallel Simulations With Asynchronous Cellular Automata, Proceedings of the 1993 European Simulation Symposium, 397-402, **1993**.

[145] S. Robinson, Simulation: The Practice of Model Development and Use, Vol. 50, 1. Baskı, Wiley, **2004**.

EKLER

Ek - 1. Tezde Kullanılan Terimlerin Türkçe – İngilizce Karşılıkları

Türkçe	İngilizce
Alana özgü	Domain specific
Alt-durum kaydetme	Sub-state saving
Anlamsal özellikler	Semantics
Artırımlı durum kaydetme	Incemental state saving
Başarım	Performance
Benzetim durumu	Simulation state
Benzetim zamanı	Simulation time
Çalışma tutumu	Execution profile
Çok büyük ölçekte koşut	Massively parallel
Çok çekirdekli	Many core
Çoklu çekirdekli	Multi-core
Çoklu-iş parçacıklı	Multi-threaded
Çoklu-sürüm alma	Multi-versioning
Dağıtılmış	Distributed
Dağıtılmış bilgisayar kümesi	Beowulf cluster
Denetim noktası	Checkpoint
Durum farkı	State difference
Durum geri yükleme	State restoring
Durum kaydetme	State saving
Durum kopyası kaydetme	Copy state saving
Düğüm	Node
Eşgüdümlü	Coordinated
Eşgüdümsüz	Uncoordinated
Etkileşim maskesi	Interaction mask
Etmen	Agent
Etmen-koşut	Agent-parallel
Fosil toplama	Fossil collection
Genel zaman (genel benzetim zamanı)	Global time (global clock)

Geri sarma	Rollback
Hücresel özdevinirler	Cellular automaton
Izgara	Grid
İleri bakma	Lookahead
İleri sarma	Coasting forward
İyimser zaman yönetimi	Optimistic time management
Karşıt ileti	Anti-message
Kendini uyarlayabilen	Self adaptive
Kesikli olay benzetimi	Discrete event simulation
Koşut	Parallel
Mantıksal İşlem	Logical process
Melez	Hybrid
Mızıkçı ileti	Straggler message
Mızıkçı ileti izi	Straggler message signature
Nedensellik	Causality
Olay	Event
Olay atama	Event scheduling
Olay işleyici	Event-handler
Olay kuyruğu	Event queue
Öngörülmüş geri sarma	Predetermined rollbacks
Örgü	Mesh
Sakıngan zaman yönetimi	Conservative time management
Seyrek denetim noktası	Periodic checkpointing
Sezgisellik	Heuristic
Tampon alan	Buffer
Tembel iptal etme	Lazy cancellation
Tembel yeniden değerlendirme	Lazy re-evaluation
Tersine hesaplama	Reverse computing
Tıkanıklık kontrolü	Congestion control
Toplamsal Artış Çarpımsal Azalış	Additive Increase Multiplicative Decrease
Tutum	Profile
Uyarlanabilir	Adaptive

Uzamsal	Spatial
Uzamsal belirsizlik	Spatial uncertainty
Uzamsal-koşut	Spatial-parallel
Yeniden oynatma	Re-execution
Yerel zaman (yerel benzetim zamanı)	Local time (local clock)
Yük dengeleme	Load balancing
Yüksek Başarımlı Hesaplama	High Performance Computing (HPC)
Zaman bükülmesi	Time Warp
Zaman uyumlama	Synchronization
Zamanlama	Scheduling
Zamansal belirsizlik	Temporal uncertainty

Ek - 2. Tezde Sıkça Kullanılan Terimlerin Tanımları

Benzetim durumu: Benzetimin o anki durum değişkenlerini kapsayan değişken kümesidir.

Benzetim zamanı: Benzetim zamanı, Fujimoto tarafından iki şekilde tanımlanmaktadır [1]:

- Benzetim tarafından fiziksel zamanı modellemek için kullanılan bir soyutlama.
- Modellenen fiziksel sistem zamanının her bir anının temsil edildiği, tamamen sıralı bir değerler kümesi.

Denetim noktası: İyimser zaman yönetim mekanizmalarında olası geri sarma işlemleri için benzetim durumunun kaydedildiği noktalardır. Denetim noktası mekanizmaları başta veri tabanı yönetim sistemleri olmak üzere bir çok farklı disiplinde de kullanılmaktadır.

Etkileşim modeli: Benzetimi yapılacak modelin anlamsal özelliklerini (İng. semantics) kullanarak etmenlerin davranışının hangi etmenler tarafından hangi koşullar altında etkileneceğini bildiren bir belirtimdir.

Genel zaman (genel benzetim zamanı): Koşut veya dağıtılmış bir benzetimde, benzetime katılan tüm mantıksal işlemler tarafından bilinen ortak benzetim zamanıdır. Genel zamanın, tüm mantıksal işlemler arasındaki en küçük yerel zaman değerine eşit olduğu söylenebilir.

Geri sarma: İyimser zaman yönetim mekanizmalarında, zaman uyumlama eksikliğinden dolayı bir mızıkçı ileti alınması durumunda, benzetimin eski bir zamana geri dönmesi sürecidir.

İleri sarma: Denetim noktası mekanizmalarını kullanan iyimser zaman yönetim mekanizmalarında bir geri sarma işlemiyle eski bir denetim noktasına dönüldükten sonra mızıkçı iletinin zamanına kadar olan eski olayların yeniden işletilmesidir.

Karşıt ileti: Bir geri sarma işlemi yapan mantıksal işlemin, başka bir mantıksal işleme hatalı bir olay atadığını fark etmesi üzerine bu hatanın telafi edilmesini sağlamak için aynı mantıksal işleme gönderdiği özel ileti.

Kesikli olay benzetimi: "Belirli bir sistemin davranışını zaman içindeki kesikli olayların bir dizisini kullanarak ifade eden bir benzetim yaklaşımıdır. Bu yaklaşımda olaylar yalnızca kesikli olarak ifade edilen bir zamanda gerçekleşerek benzetim durumunda bir değişiklik yaratırlar [83][145]".

Mantıksal işlem: Koşut veya dağıtılmış bir benzetime katılarak benzetimin bir bölümünü gerçekleştirme görevini üstlenen işlemdir. Mİ'ler birbirine koşut olarak çalışıp benzetimi birlikte yürütürler.

Mızıkçı ileti: İyimser zaman yönetim mekanizmalarında, zaman uyumlama eksikliğinden dolayı bir mantıksal işlemin yerel zamanı, diğer mantıksal işlemlerin yerel zamanlarından ileride olabilir. Böyle bir durumda bir mantıksal işlem, yerel zamanı daha ileride olan bir mantıksal işleme olay atadığında, bu olayı alan mantıksal işlem geçmiş zaman damgalı olayı içeren iletiyi mızıkçı ileti olarak tanımlar.

Olay: Ortaya çıktığında, benzetimin bir dizi işlem yaparak yerine getirdiği eylemdir. Koşut veya dağıtılmış bir benzetimi çalıştıran mantıksal işlemler, kendilerine olay atayabilecekleri gibi birbirlerine de olay atayabilirler.

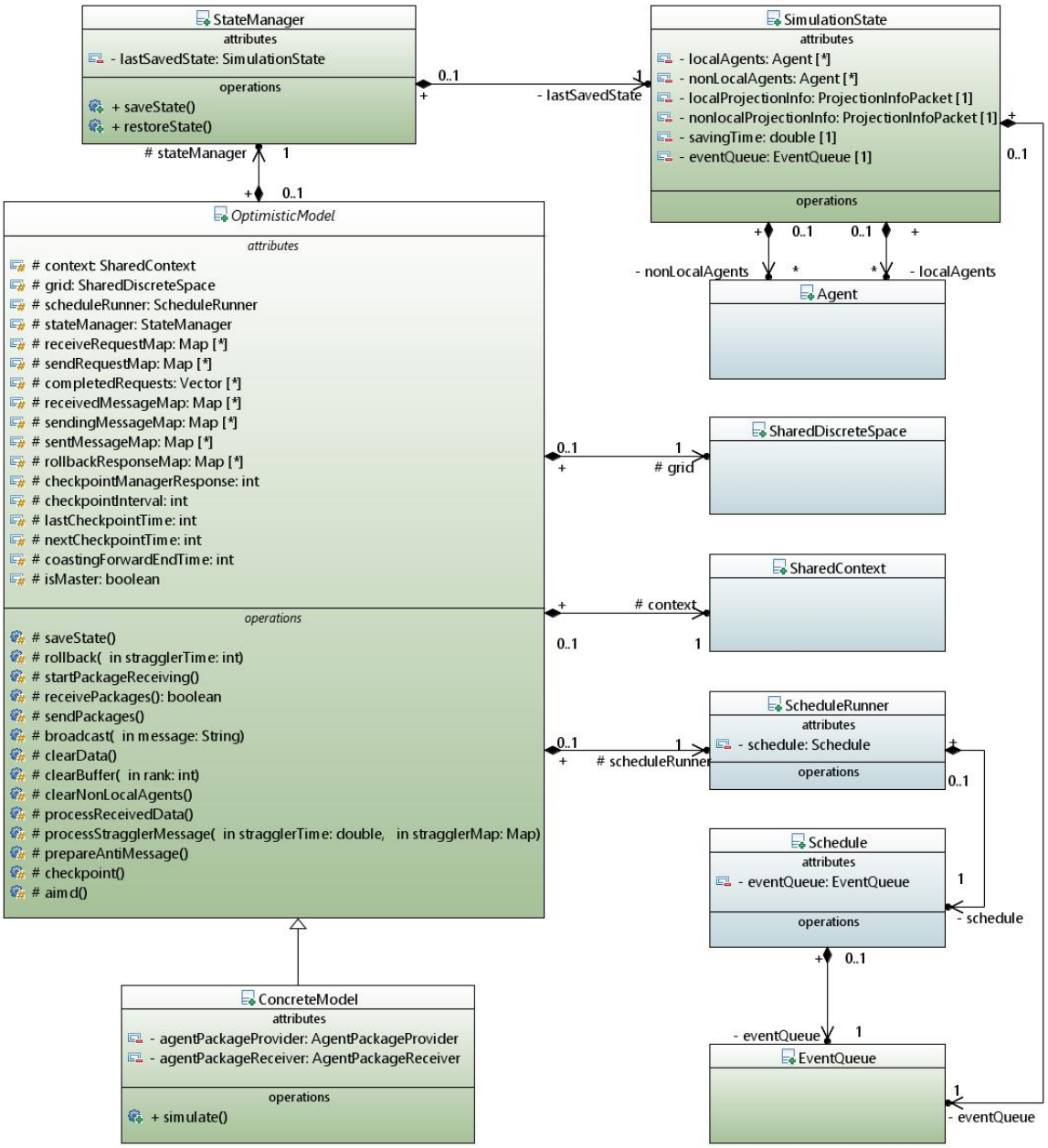
Olay kuyruğu: Bir mantıksal işlemin ileri zamanda işlenecek olayları tuttuğu veri yapısıdır.

Tampon alan: Bir mantıksal işlemin sınır bölgelerinde yer alan ve mantıksal işlemin diğer mantıksal işlemlerle zaman uyumlama yapmasını gerektiren uzamsal alandır.

Yerel zaman (yerel benzetim zamanı): Koşut veya dağıtılmış bir benzetimde, benzetime katılan bir mantıksal işlemin kendi içerisinde yönettiği benzetim zamanıdır.

Ek - 3. RHPC_TW Yazılımına Ait Sınıf Şeması ve Ayrıntıları

Bölüm 5'te ayrıntıları verilen Zaman Bükülmesi mekanizmasıyla genişletilmiş Repast HPC (RHPC_TW) yazılımının mimarisi sınıf şeması üzerinden özet bir şekilde aşağıda verilmiştir. Bu şemada Repast HPC'ye ait olan ve iyimser zaman yönetim mekanizmasını ilgilendirmeyen kısımlar ile RHPC_TW için geliştirilen yardımcı sınıflara yer verilmemiştir.



Bu şemada yeşil renk ile gösterilen sınıflar (OptimisticModel, ConcreteModel, EventQueue, StateManager ve SimulationState) RHPC_TW için geliştirilmiş olan sınıfları; mavi renk ile gösterilen sınıflar (Agent, SharedDiscreteSpace, SharedContext, ScheduleRunner ve Schedule) ise Repast HPC'de bulunan ve RHPC_TW için genişletilen sınıflardan bazılarını göstermektedir. Bahsi geçen bu sınıfların ayrıntıları aşağıda verilmiştir.

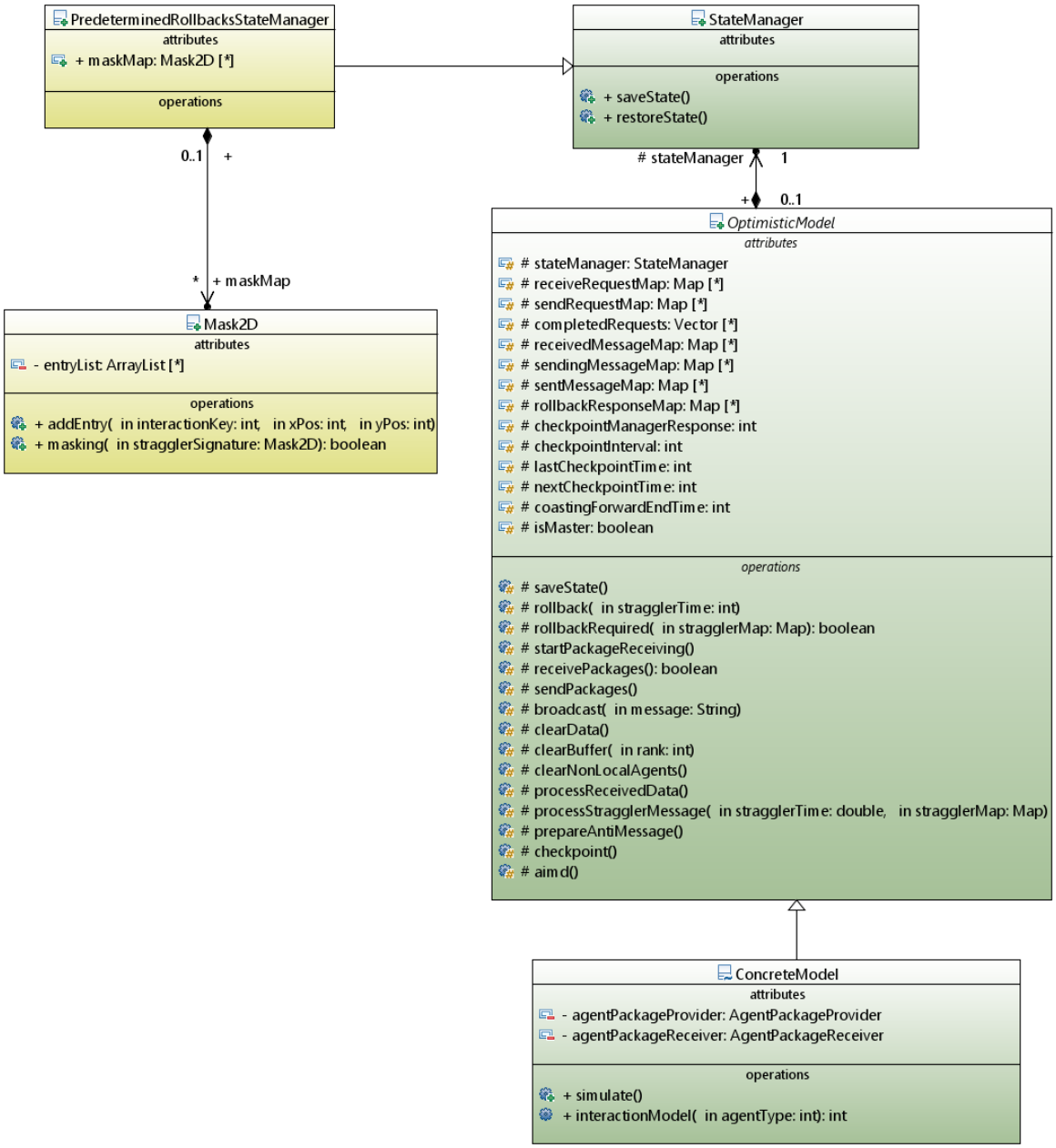
RHPC_TW için Geliştirilen Sınıf	**İşlevsellik**
OptimisticModel	Repast HPC, RepastProcess sınıfındaki işlevler yardımıyla Mİ'lerin sakıngan zaman yönetim mekanizmasına göre zaman uyumlamasını sağlar. RHPC_TW'de bu görev OptimisticModel sınıfına verilmiştir. Bu sınıf geliştiriciden soyutlanmış olup, geliştiricinin bu sınıf hakkında bilgiye sahip olması gerekmemektedir. Geliştiricinin tek yapması gereken, OptimisticModel sınıfından türetilen ConcreteModel sınıfı içerisinde tıpkı Repast HPC'de olduğu gibi modelini gerçekleştirmesidir. OptimisticModel'in işlevleri şu şekilde özetlenebilir: • Zaman yönetim mekanizmasının, etmenlerin yaşadığı benzetim ortamı üzerinden yönetilmesi, • Geri sarma işlemlerinin yönetilmesi, • Komşu mantıksal işlemlerden alınan iletilerin modele uygun bir biçimde işlenmesi, • Karşıt iletilerin yönetilmesi, • Denetim noktası alınması, benzetim durumlarının kaydedilmesi – geri yüklenmesi ve bir sonraki denetim noktasının ne zaman alınacağının belirlemesi.

RHPC_TW için Geliştirilen Sınıf	**İşlevsellik**
ConcreteModel	Bu sınıf, OptimisticModel'den türetilmekte olup, geliştiricinin ihtiyaçlarına göre modelini tanımlaması amacıyla kullanılmaktadır. Ayrıca Repast HPC'de olduğu gibi, mantıksal işlemler arasındaki iletilerin oluşturulması için gerekli olan etmen paketi alıcı (agentPackageProvider) ve etmen paketi sağlayıcı (agentPackageReceiver) yapılarının da bu sınıf içerisinde tanımlanması gerekmektedir.
EventQueue	Mantıksal işlemlerin ileriye dönük olarak zamanlanmış olay bilgilerini tutması için bu veri yapısı kullanılmaktadır. Ayrıca benzetimdeki olaylar önceliklerine göre belli türlere ayrılmıştır. Olay türleri, önceliği en düşük olandan en yüksek olana doğru şu şekilde tanımlanmıştır: • STOP (benzetim tamamlandığında çalıştırılması planlanan olaylar) • CHECKPOINT (denetim noktası alma olayları) • LAZY (önceliği düşük olaylar) • NORMAL (varsayılan öncelikli olaylar) • IMMEDIATE (önceliği yüksek olaylar)
StateManager	Denetim noktası alma işlemleri sırasında benzetim durumlarının kaydedilmesi ve geri sarma işlemleri sırasında eski benzetim durumlarının geri yüklenmesi işlevleri bu sınıfın sorumluluğundadır. Bu sınıf, sadece en son denetim noktasında kaydedilen benzetim durumunu tutar; çoklu-sürüm alma tekniğini kullanmaz.

RHPC_TW için Geliştirilen Sınıf	**İşlevsellik**
SimulationState	Mantıksal işlemler benzetim durumlarını bu veri yapısı içerisine kaydeder. Bu benzetim durumu içerisinde o anki; • Zaman bilgisi, • Sahip olunan yerel etmenler, • Diğer mantıksal işlemlerden alınmış olan yerel olmayan etmenler, • Etmenlerin yaşadığı ortam bilgileri ve • Olay kuyruğunda yer alan ve ileriye dönük olarak zamanlanmış olay bilgileri yer alır.
Agent	Bu sınıf halihazırda Repast HPC'de kullanılan ve etmenleri tanımlamak için yararlanılan veri yapısıdır. Repast HPC, etmenleri birbirinden ayırmak için her bir etmen özgün (İng. *unique*) bir AgentID ile eşleştirir. Bir AgentID, `id`, `startingRank`, `agentType` ve `currentRank` bilgilerini tutar.
SharedDiscreteSpace	Bu sınıf halihazırda Repast HPC'de kullanılan ve etmenlerin üzerinde yaşadığı 2 boyutlu uzayı tanımlamak için kullanılan veri yapısıdır.
SharedContext	Bu sınıf halihazırda Repast HPC'de kullanılan ve çok sayıdaki uzayı tutan veri yapısıdır.
ScheduleRunner	Bu sınıf halihazırda Repast HPC'de kullanılmakta ve benzetim zamanını ilerleten mekanizmayı sağlamaktadır. RHPC_TW'de zaman yönetim mekanizmasının iyimser yaklaşımla iletilmesi sorumluluğu bu sınıfa verilmiştir.
Schedule	Bu sınıf halihazırda Repast HPC'de kullanılmakta ve benzetim zamanını tutmaktadır. RHPC_TW'de tanımlanan olay kuyruğunu (EventQueue) yönetme işi de bu sınıfın sorumluluğundadır.

Ek - 4. RHPC_PR Yazılımına Ait Sınıf Şeması ve Ayrıntıları

Bölüm 7'de ayrıntıları verilen Öngörülmüş Geri Sarma mekanizmasıyla genişletilmiş Repast HPC (RHPC_PR) yazılımının mimarisi sınıf şeması üzerinden özet bir şekilde aşağıda verilmiştir. Bu şemada Ek - 3'te verilen RHPC_TW sınıf şemasını genişleten sınıflara yer verilmiştir.



Bu şemada yeşil renk ile gösterilen sınıflar (OptimisticModel ve StateManager) RHPC_TW için geliştirilmiş olan sınıfları; sarı renk ile gösterilen sınıflar

(PredeterminedRollbacksStateManager ve Mask2D) ise Öngörülmüş Geri Sarma mekanizması için geliştirilmiş sınıflardan bazılarını göstermektedir. Bahsi geçen bu sınıfların ayrıntıları aşağıda verilmiştir.

RHPC_TW için Geliştirilen Sınıf	**İşlevsellik**
OptimisticModel	RHPC_TW'dekine ek olarak OptimisticModel sınıfı; bir mantıksal işlem bir mızıkçı ileti aldığında mızıkçı ileti izi oluşturarak geri sarma işlemi yapılıp yapılmayacağına karar veren maskeleme işlevini çağırmayı sağlar.
ConcreteModel	RHPC_TW'dekine ek olarak ConcreteModel sınıfı; geliştiricinin etkileşim modelini tanımlayabileceği interactionModel işlevinin yazılmasına olanak sağlar. Bu işlev parametre olarak aldığı etmen türü değerine göre bir asal sayıyı üretir. Bu asal sayı, verilen etmen türünün etkileşime girebileceği etmen türlerini ifade eder ve etkileşim maskesine dahil olur. Etkileşim maskesinin nasıl oluşturulduğu ile ilgili ayrıntılı bilgi Bölüm 7.2'de verilmiştir. Model geliştirici interactionModel işlevini tanımlamak zorunda değildir. Ancak; eğer bu işlev tanımlanmazsa, Öngörülmüş Geri Sarma mekanizması Bölüm 7.1'de verildiği gibi yalnızca etmenlerin benzetim uzayındaki konumunu göz önünde bulundurarak çalışacaktır.

RHPC_TW için Geliştirilen Sınıf	İşlevsellik
Mask2D	Öngörülmüş Geri Sarma mekanizmasının kullandığı etkileşim maskeleri için geliştirilmiş veri yapısıdır. Ayrıca bir etkileşim maskesinin mızıkçı ileti izi ile maskelenmesi de bu sınıfın sorumluluğundadır.
PredeterminedRollbacksStateManager	Etkileşim maskelerinin kaydedilmesini ve yönetilmesini sağlayan sınıftır. Etkileşim maskeleri, benzetim durumu göz önünde bulundurularak oluşturulur; bu yüzden StateManager sınıfının bir alt sınıfı olarak gerçekleştirilmiştir.

Ek - 5. Tezden Türetilmiş Yayınlar

[1] **B. K. Görür**, K. M. İmre, H. Oğuztüzün, L. Yılmaz, Predetermined Rollbacks: An Extension to Time Warp for Spatially Parallel Agent-Based Simulation, Simulation Modelling Practice and Theory, 95(**2019**), 60-77
DOI: 10.1016/j.simpat.2019.04.008.

Ek - 6. Tezden Türetilmiş Bildiriler

[1] **B. K. Görür**, K. M. İmre, H. Oğuztüzün, L. Yılmaz, Repast HPC with Optimistic Time Management, Proceedings of the 24th High Performance Computing Symposium (SpringSim, HPC '16), 3-6 Nisan, San Diego, California, USA, **2016**.
DOI: 10.22360/SpringSim.2016.HPC.046,
URI: https://dl.acm.org/citation.cfm?id=2972973

[2] **B. K. Görür**, K. M. İmre, H. Oğuztüzün, L. Yılmaz, Improving the Performance of Optimistic Time Management Mechanism with Sub-State Saving, Proceedings of the 25th High Performance Computing Symposium (SpringSim, HPC '17), 23-26 Nisan, Virginia Beach, Virginia, USA, **2017**.
DOI: 10.22360/springsim.2017.hpc.035
URI: https://dl.acm.org/citation.cfm?id=3108107

Ek - 7. Tez Çalışması Orijinallik Raporu

HACETTEPE ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
~~YÜKSEK LİSANS~~/DOKTORA TEZ ÇALIŞMASI ORJİNALLİK RAPORU

HACETTEPE ÜNİVERSİTESİ
FEN BİLİMLER ENSTİTÜSÜ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI BAŞKANLIĞI'NA

Tarih: 03/05/2019

Tez Başlığı: UZAMSAL KOŞUT VE DAĞITILMIŞ BENZETİMLER İÇİN İYİMSER BİR ZAMAN YÖNETİM MEKANİZMASI

Yukarıda başlığı gösterilen tez çalışmamın a) Kapak sayfası, b) Giriş, c) Ana bölümler d) Sonuç kısımlarından oluşan toplam 171 sayfalık kısmına ilişkin, 02/05/2019 tarihinde tez danışmanım tarafından *Turnitin* adlı intihal tespit programından aşağıda belirtilen filtrelemeler uygulanarak alınmış olan orijinallik raporuna göre, tezimin benzerlik oranı % 4 'tür.

Uygulanan filtrelemeler:
1- Kaynakça hariç
2- Alıntılar hariç
3- 5 kelimeden daha az örtüşme içeren metin kısımları hariç

Hacettepe Üniversitesi Fen Bilimleri Enstitüsü Tez Çalışması Orjinallik Raporu Alınması ve Kullanılması Uygulama Esasları'nı inceledim ve bu Uygulama Esasları'nda belirtilen azami benzerlik oranlarına göre tez çalışmamın herhangi bir intihal içermediğini; aksinin tespit edileceği muhtemel durumda doğabilecek her türlü hukuki sorumluluğu kabul ettiğimi ve yukarıda vermiş olduğum bilgilerin doğru olduğunu beyan ederim.

Gereğini saygılarımla arz ederim.

03.05.2019
Tarih ve İmza

Adı Soyadı: Bilge Kaan GÖRÜR
Öğrenci No: N10162649
Anabilim Dalı: Bilgisayar Mühendisliği
Programı: Bilgisayar Mühendisliği-Bütünleşik Doktora
Statüsü: ☐ Y.Lisans ☐ Doktora ☒ Bütünleşik Dr.

DANIŞMAN ONAYI

UYGUNDUR.

Doç.Dr. Kayhan İMRE

(Unvan, Ad Soyad, İmza)

Ek - 8. Özgeçmiş

Kişisel Bilgiler

Adı Soyadı	:	Bilge Kaan Görür
Doğum Yeri	:	Kayseri
Doğum Tarihi	:	1988
Medeni Hali	:	Evli
E-posta	:	bkaangorur@gmail.com

Eğitim

Lise	:	Kayseri Fen Lisesi (2002-2005)
Lisans	:	Hacettepe Üniversitesi Bilgisayar Mühendisliği, 3,05/4,00 (2005-2010)
Yüksek Lisans	:	-
Doktora	:	Hacettepe Üniversitesi Bilgisayar Mühendisliği 3,79/4,00 (2010-2019)

Yabancı Dil ve Düzeyi

İngilizce – İyi

İş Deneyimi

Eylül 2010 – Şubat 2016	:	Araştırma Görevlisi, Orta Doğu Teknik Üniversitesi Türk Silahlı Kuvvetleri Modelleme ve Simülasyon Araştırma ve Uygulama Merkezi
Şubat 2016 – Temmuz 2017	:	Uzman Mühendis, ROKETSAN A.Ş.
Temmuz 2017 – …	:	Kıdemli Uzman Mühendis, ROKETSAN A.Ş.

Deneyim Alanları

Modelleme ve benzetim, etmen tabanlı modelleme ve benzetim, koşut ve dağıtılmış sistemler, benzetim görselleştirme, model güdümlü geliştirme, makine öğrenmesi.

Yayınlar

[1] A. Yılmaz, D. Yılmaz, A. M. Şenyiğit, **B. K. Görür**, V. İşler, Genel Amaçlı Araştırma Simülatörü: Donanım ve Yazılım Altyapısının Tasarlanması ve Geliştirilmesi, Dördüncü Ulusal Savunma Uygulamaları Modelleme ve Simülasyon Konferansı (USMOS 2011), 14-15 Haziran, Ankara, Türkiye, **2011.**

[2] A. Yılmaz, D. Yılmaz, A. M. Şenyiğit, **B. K. Görür**, V. İşler, Genel Amaçlı Araştırma Simülatörü: Donanım ve Yazılım Altyapısının Tasarlanması ve Geliştirilmesi, Savunma Bilimleri Dergisi 11 (**2011**) 1 (USMOS 2011 özel sayısı)**.**

[3] J. Ledet, A. Teran-Somohano, Z. Butcher, L. Yılmaz, A. E. Smith, H. Oğuztüzün, O. Dayıbaş, **B. K. Görür**: Toward model-driven engineering principles and practices for model replicability and experiment reproducibility. Proceedings of the Symposium on Theory of Modeling & Simulation-DEVS Integrative (SpringSim, TMS-DEVS'2014), 13-16 Nisan, Tampa, Florida, USA, **2014**.

[4] J. Ledet, S. Çam, **B. K. Görür**, O. Dayıbaş, H. Oğuztüzün, L. Yılmaz, A. E. Smith. A Hybrid Transformation Process for Simulation Modernization and Reuse via Model Replicability and Scenario Reproducibility, 2015 Alabama International Simulation Conference, Mayıs, Huntsville, Alabama, USA, **2015**.

[5] **B. K. Görür**, K. M. İmre, H. Oğuztüzün, L. Yılmaz, Repast HPC with Optimistic Time Management, Proceedings of the 24th High Performance Computing Symposium (SpringSim, HPC '16), 3-6 Nisan, San Diego, California, USA, **2016**.

[6] **B. K. Görür**, A. A. Muhiddinoğlu, K. Yayla, A. N. Çallı, Mühendislik Benzetimleri için Genel Amaçlı Görselleştirme Alt Yapısı: Mimari Tanım ve Durum Çalışması, 8. Savunma Teknolojileri Kongresi (SAVTEK 2016). 12-14 Ekim, Ankara, Türkiye, **2016**.

[7] **B. K. Görür**, K. M. İmre, H. Oğuztüzün, L. Yılmaz, Improving the Performance of Optimistic Time Management Mechanism with Sub-State Saving, Proceedings of the 25th High Performance Computing Symposium (SpringSim, HPC '17), 23-26 Nisan, Virginia Beach, Virginia, USA, **2017**.

[8] **B. K. Görür**, A. N. Çallı. Semi-Automatic Parallelization of Simulations with Model Transformation Techniques, Proceedings of the Symposium on Model-driven Approaches for Simulation Engineering (SpringSim, Mod4Sim'17), 23-26 Nisan, Virginia Beach, Virginia, USA, **2017**.

[9] Ö. Demirtaş, **B. K. Görür**. Koşut ve Dağıtık Monte Carlo Koşturucusu. Yedinci Ulusal Savunma Uygulamaları Modelleme ve Simülasyon Konferansı (USMOS 2017), 21-23 Kasım, Ankara, Türkiye, **2017**.

[10] L. Yilmaz, S. Chakladar, K. Doud, A. E. Smith, A. Teran-Somohano, H. Oğuztüzün, S. Çam, O. Dayıbaş, **B. K. Görür**. Models as self-aware cognitive agents and adaptive mediators for model-driven science, 2017 Winter Simulation Conference, 3-6 Aralık, Las Vegas, NVA, USA, **2017**.

[11] **B. K. Görür**, A. N. Çallı. An Object Oriented Agent Based Framework for Modeling and Simulation in Aerospace, 2018 AIAA Modeling and Simulation Technologies Conference – AIAA Scitech Forum, 8-12 Ocak, Kissimmee, Florida, USA, **2018.** DOI: 10.2514/6.2018-0119.

[12] S. Çam, O. Dayıbaş, **B. K. Görür**, H. Oguztüzün, L. Yilmaz, S. Chakladar, K. Doud, A. E. Smith, A. Teran-Somohano, Supporting Simulation Experiments with Megamodeling, Proceedings of the 6th International Conference on Model-Driven Engineering and Software Development - Volume 1: MODELSWARD, 22-24 Ocak, Funchal, Madeira, Portugal, **2018**.

[13] **B. K. Görür**, K. M. İmre, H. Oğuztüzün, L. Yılmaz. Predetermined Rollbacks: An Extension to Time Warp for Spatially Parallel Agent-Based Simulation. Simulation Modelling Practice and Theory, Volume 95, September, **2019.** DOI: 10.1016/j.simpat.2019.04.008.

Diğer Akademik Etkinlikler

- Dergi hakemliği: Simulation Modelling Practice and Theory (SIMPAT)
- Alt konferans eş-başkanlığı (İng. *track co-chair*): Summer Simulation Multi Conference 2018, Work-in-Progress Track
- Konferans hakemliği: Spring Simulation Multi-Conference
- Konferans hakemliği: Summer Simulation Multi-Conference
- Konferans hakemliği: AIAA Science and Technology Forum (SciTech), Modeling and Simulation Technologies

- Editörlük: Dördüncü Ulusal Savunma Uygulamaları, Modelleme ve Simülasyon Konferansı Bildiri Kitabı (USMOS 2011).
- Editör Yardımcılığı: Savunma Bilimleri Dergisi 11 (**2011**) 1 (USMOS 2011 özel sayısı).