Sistem Algoritmaları Modelleme, Simülasyon ve Analiz Aracı (SAMSA)

Burak KEKEÇ[1] Mustafa CANAY[2]

[1,2]Radar ED\ET Sistem Mühendisliği Müdürlüğü, Aselsan, Ankara

[1]e-posta: bkekec@aselsan.com.tr

[2]e-posta: mcanay@aselsan.com.tr

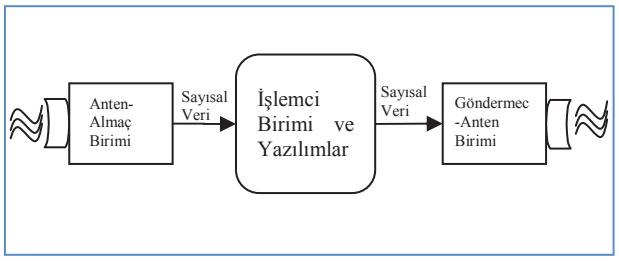
Özet. Sistem Algoritmaları Modelleme, Simülasyon ve Analiz Aracı (SAMSA), farklı dillerde gerçeklenmiş, girdisi ve çıktısı belli olan algoritma bileşenlerini birbirlerine bağlayarak bir sistem **modellemesi** yapmaya olanak sağlayan bir yazılımdır. SAMSA kullanılarak oluşturulan model, algoritmalara gelebilecek girdiler ile beslenerek sistemin çalışma zamanındaki davranışının **simülasyonu** yapılabilmektedir. Ayrıca gerek her bir algoritma bileşenin gerekse oluşturulan modelin bütününün çıktıları incelenerek ve karşılaştırılarak **analiz** işlemi yapılmasına olanak sağlamaktadır. Böylece algoritmaların tek tek çalışmaları doğrulanabildiği gibi bir arada çalışmaları sonucunda ortaya çıkacak sistem performansı da tasarımın ön safhalarında gözlemlenmiş olabilmektedir.

Anahtar Kelimeler: Sistem, Modelleme, Simülasyon, Analiz

1 Giriş

Modelleme, simülasyon ve analiz mühendislik alanındaki araştırmalarda ve kaliteli sistemler geliştirmede kullanılan çok önemli yöntemlerdir.[1] Sistem Algoritmaları Modelleme, Simülasyon ve Analiz Aracı (SAMSA) Radar Elektronik Harp alanında bu tekniklerin uygulanması amacıyla geliştirilmiş bir yazılımdır.

Radar Elektronik Harp sistemleri genel olarak anten-almaç donanım birimleri, işlemci birimleri ve üzerlerinde çalışan yazılımlar ve sistemin türüne göre gerekiyorsa göndermeç-anten donanım birimlerinden oluşmaktadır. Anten-almaç birimleri sinyallerin sistem tarafından alınmasını ve sayısallaştırılmasını sağlamaktadır. Benzer şekilde göndermeç-anten birimleri de istenilen özelliklerde sinyal üretilmesi ve yayılması işlemini yapmaktadır. Bu iki donanım birimi arasında işlemci birimleri ve üzerlerinde çalışan yazılımlar bulunmaktadır. Radar Elektronik Harp projelerindeki genel birimler Şekil 1'deki gibidir.



Şekil 1. Radar Elektronik Harp Sistemlerindeki Temel Birimler

Radar Elektronik Harp sistemlerinde bulunan yazılımlar üç farklı türdedir. Birinci tür yazılımlar, farklı amaçlarla tasarlanmış sinyal işleme veya veri işleme algoritmalarının gerçeklemeleridir. İkincisi, algoritma yazılımlarının bir arada nasıl çalışacaklarını tanımlayan senaryoları işleten ve donanım birimlerini kontrol eden kontrol yazılımlarıdır. Üçüncü tür ise sistem ile sistemi kullanan operatörün etkileşimini sağlayan kullanıcı arayüzü yazılımlarıdır.

Radar Elektronik Harp sistemlerinde çalışan algoritmalar sistemin performansını en çok etkileyen faktördür. Bu nedenle algoritmaların, sistemin çalışacağı ortama uygun olarak amacını gerçekleştirecek en iyi tasarıma sahip olması önemlidir. Ayrıca bir algoritmanın çıktısı diğer algoritmanın girdisi olacağından bu algoritmaların birbiriyle uyum içinde çalışması da sistem performansı için çok önemlidir. Bu nedenle algoritmaların bütününün ele alınması ve bu bütünün oluşturduğu yapının iyi tasarlanması ve test edilmesi gerekmektedir. Ancak bu yapıyı oluşturan algoritma parçaları farklı platformlarda farklı dillerde gerçeklenmiş olabilmektedir. SAMSA yazılımıyla, bu algoritma parçalarını aynı platform üzerinde birlikte çalıştırılacak şekilde bir modelleme yapılabilmekte, bu modelin çalışma zamanındaki davranışını simüle edebilmekte ve bu modelin çalıştırılması ile üretilen çıktı analiz edilebilmektedir.

Bildirinin 2. bölümünde SAMSA'nın tasarımı, 3. bölümünde SAMSA'nın yetenekleri ve faydaları anlatılacaktır.

2 SAMSA'nın Tasarımı

SAMSA, algoritma bileşenlerini, bileşenlerin arasındaki bağlantıları tanımlayan bir modele sahiptir. Ayrıca SAMSA ile oluşturulan bir sistem modelini işleten bir senaryo kontrol bileşeni içermektedir.

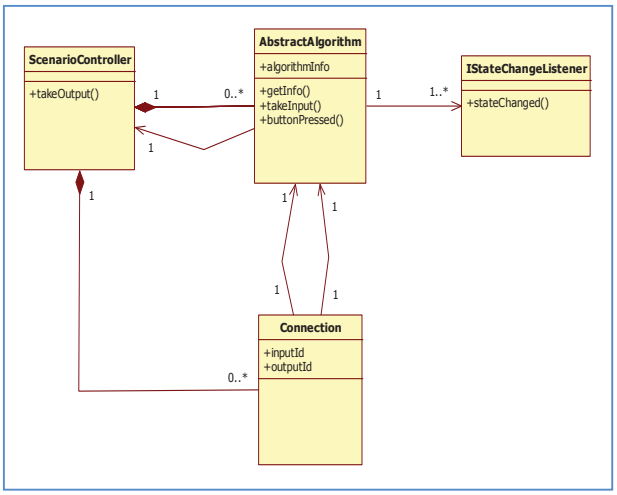
Bu modele uygun bir şekilde SAMSA'ya C/C++, Matlab, Java gibi dillerde gerçeklenmiş yeni algoritma bileşenleri eklenebilmektedir. Matlab'ta yazılmış bir bileşenin eklenmesi için Matlab Java Builder[2] eklentisi kullanılmaktadır. C/C++ ile geliştirilmiş algoritmaların için Java JNI altyapısı[3] kullanılmaktadır.

SAMSA yazılımında, özel bir işlevi yerine getirmek için tasarlanmış algoritma bileşenlerinin dışında ortak kullanım amaçlı önceden tanımlı algoritma bileşenleri bulunmaktadır. Bu bileşenler genel amaçlı oldukları için girdi ve çıktının tipinden bağımsız

çalışacak şekilde tasarlanmıştır. Bu soyutlama Java'nın yansıma (reflection) altyapısı[4] ile sağlanmıştır.
Ayrıca aracın kullanımını kolaylaştırmak amacıyla mümkün olduğunca sade bir kullanıcı arayüzü tasarlanmış ve bu arayüz ile sistem modelleme yapma, modelin simülasyonunu ve analizini yapma imkanı sağlanmıştır.

2.1 Algoritma Modeli

SAMSA'ya eklenecek olan her bir algoritma bileşeni SAMSA algoritma modeline uygun bir şekilde geliştiriyor olmalıdır. Her bir algoritma bileşeninin Java dilinde gerçeklenmiş bir algoritma kabuğu olmalıdır. Bu algoritma kabuğunun içerisinden ilgili algoritma geliştirildiği platforma göre çalıştırılır.
SAMSA algoritma modeli, SAMSA içerisindeki algoritmaların yapılarını ve birbirleri arasındaki bağlantıları tanımlar. Bu modeli tanımlayan sınıflar ve bu sınıfların birbirleriyle ilişkileri Şekil 2'teki gibidir.



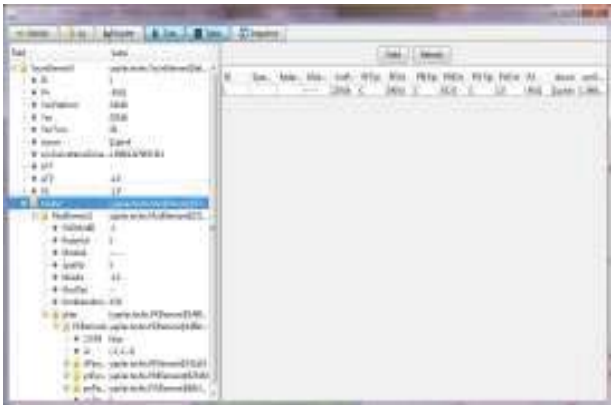
Şekil 2. SAMSA Algoritma Modeli

SAMSA'ya eklenecek her bir algoritma bileşeninin *AbstractAlgorithm* sınıfını genişletiyor olması gerekmektedir. Bu sınıfta bulunan *algorithmInfo* alanı algoritmanın adını, girdilerini çıktılarını ve hakkında bilgilerini içermektedir. Bu alan her bir algoritma bileşeni için doldurulmalıdır. Her bir algoritma için kullanıcı girişi alındığında yapılacak işlemleri tanımlamak için *buttonPressed* metotunun gerçeklenmesi gerekmektedir. Bir algoritma bileşeninin çalışması ilgili bileşenin *takeInput* metotunun çağırılması ile gerçekleştirilir. Bu metot içerisinde gelen girdinin türüne göre algoritma yapması gereken işlemleri gerçekleştirecektir. Algoritma bileşenleri çalışma ve veri bekleme durumları arasındaki geçişleri *IStateChangeListener* olarak kendine kayıt olmuş bileşen-

lere aktarmaktadır. Bu bilgi ile sistemde hangi algoritmanın ne kadar çalıştığı gibi bilgiler elde edilebilmektedir.
Connection sınıfı hangi algoritmanın hangi çıktısının hangi algoritmanın hangi girdisiyle bağlı olduğu bilgisini barındırmaktadır.
ScenarioController sınıfı algoritmaların sırasıyla çalıştırılmasını sağlar. Çalışmasını bitirip çıktısını üreten algoritma bu sınıfa çıktısını aktarır. *ScenarioController* sınıfı da *Connection* sınıflarında tanımlanan bağlantılara göre ilgili çıktıyı girdi olarak bekleyen algoritmaları çalıştırır.

2.2 SAMSA'daki Genel Algoritma Bileşenleri

Veri Editörü Bileşeni .
Veri editörü bileşeni, girdisi ve çıktısı *Object* tipinde olan bir bileşendir. Bu bileşen bağlı olduğu algoritmanın çıktısını alarak tablolu ağaç (treetable) yapısında veya tablo olarak kullanıcıya bir arayüz yardımıyla sunan bir bileşendir. Bunu sağlamak için Java yansıma altyapısı kullanılmıştır. Veri editörü bileşeni, yansıma altyapısı sayesinde girdi olarak gelen objenin sınıfına, sınıfın alanlarının isimlerine ve değerlerine erişerek arayüzde sunmaktadır. Bu arayüz Şekil 3' teki gibi bir görünüme sahiptir.



Şekil 3. Veri Editörü Kullanıcı Arayüzü Görünümü

Yine bu bileşen kullanılarak bu bileşene gelen veriler içerisinden seçim yapılarak bir sonraki algoritma bileşenine veri aktarılabilmektedir. Ayrıca gelen veriler kaydedilip tekrar yüklenebilmektedir. Böylece bir senaryoya aradan girilerek senaryonun kalan kısmı daha önceden kaydedilmiş bir veriyle beslenebilir.
Bir sistem modelinde her bir algoritmanın çıktısına bir veri editörü bağlanarak sistemin çalışması ara basamaklarda kontrol edilebilmektedir.

Veri Yazıcı Bileşeni.
Veri yazıcı bileşeni *Object* tipinde girdi almaktadır ama herhangi bir çıktı üretmemektedir. Kendisine gelen girdiyi veri editörü gibi yansıma altyapısı ile keşfedip belirli bir formatta konsola yazdırmaktadır.

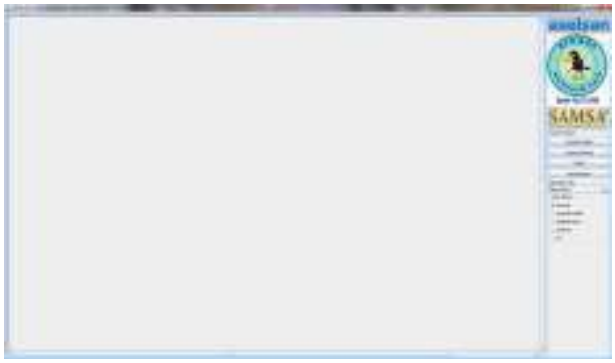
Filtre Bileşeni.
Filtre bileşeni kendisine gelen veriyi daha önceden belirlenmiş kurallara göre filtreler ve filtrelenmiş veriyi çıktı olarak üretir. Bu bileşen de yansıma altyapısını kullandığı için girdinin tipinden bağımsız çalışabilmektedir.

Veri Biriktirme Bileşeni.
Veri biriktirme bileşeni, kendisine girdi olarak gelen verileri liste yapısı ile peş peşe ekleyerek biriktirir. Böylece bir algoritma bileşeninin ürettiği çıktı kaybedilmeksizin biriktirilebilmektedir.

Karşılaştırma Bileşeni.
Karşılaştırma bileşeni, *Object* türünde iki girdisi vardır. Bu iki girdi, yansıma altyapısı kullanılarak alan alan karşılaştırılır ve sonuç raporlanır. Aynı işi yaptığı düşünülen iki algoritma bileşeninin veya bir algoritmanın iki farklı sürümünün çıktılarının karşılaştırması açısından faydalı olmaktadır.

2.3 Kullanıcı Arayüzü Tasarımı

SAMSA kullanıcı arayüzü, algoritmaların iç detaylarını bilmeden sadece algoritmaların girdi ve çıktılarını bilerek kolay bir şekilde sistem modeli üretebilmek amacıyla olabildiğince sade bir şekilde tasarlanmıştır. Kullanıcı arayüzünün bir görünümü Şekil 4'te verilmiştir.



Şekil 4. SAMSA Aracı Kullanıcı Arayüzü Görünümü

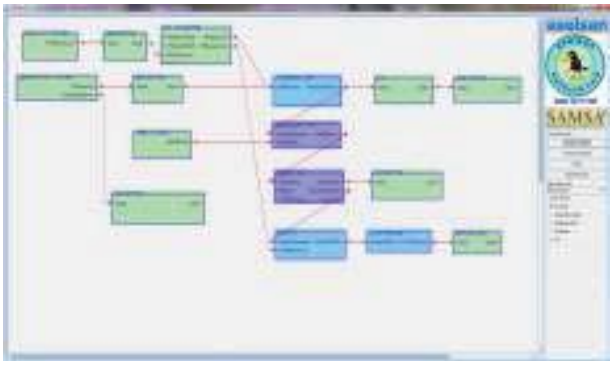
Aracın kullanıcı arayüzünün büyük bir kısmı sistem modelinin oluşturulacağı çizim alanından oluşmaktadır. Bu alanda algoritma ekleme çıkarma ve bağlantı ekleme

silme işlemleri yapılarak sistem senaryosu belirlenir. Bu alana eklenmiş her bir algoritma üzerinde kullanıcı etkileşimi sağlanmıştır.
Arayüzün sağ tarafındaki kontrol paneli ile algoritma seçimi yapılmakta ve çizim modu belirlenmektedir. Çizim modu algoritma ekleme, çıkarma, bağlantı ekleme çıkarma gibi işlemlerini tanımlar.

3 SAMSA'nın Yetenek ve Faydaları

3.1 Modelleme

SAMSA aracı, C\C++,Matlab, Java gibi dillerde gerçeklenmiş algoritma bileşenlerini barındırmaktadır. Kullanıcı, girdisi ve çıktısı belli olan bu algoritma bileşenlerini ekleyerek ve aralarındaki bağlantıları belirleyerek oluşturmak istediği sistem modelini oluşturur. Şekil 5'te örnek bir sistem modeli görüntülenmektedir.



Şekil 5. Örnek Sistem Modeli

SAMSA aracının modelleme yeteneği ile algoritma bileşenlerinin iç detaylarına hakim olmayı gerektirmeksizin bir sistem modeli oluşturmak mümkündür. Bu modelleme işlemi ile algoritmaların birbirleri ile ilişkileri ve sistem üzerindeki konumları görsel olarak sunulmuş olmaktadır. Oluşturulan model SAMSA aracıyla bir doküman oluşturularak raporlanabilmektedir.
SAMSA aracıyla oluşturulmuş bir sistem modeli xml formatında bir dosyada saklanabilmekte ve daha sonra tekrar yüklenebilmektedir.
SAMSA'da yer alan her bir algoritma bileşeninin arayüzden kullanıcı tarafından gelecek etkileşimlerde neler yapacağı algoritma özelinde tanımlanmaktadır. Örneğin algoritmaya özel konfigürasyon ayarları bu şekilde yapılabilmektedir.

3.2 Simülasyon

SAMSA yazılımı, algoritmaların her birinin ayrı ayrı testleri yapılıyor olsa da hepsinin bir arada olduğu, gerçek çalışma şeklini simüle eden bir altyapı sunmaktadır. Böylece algoritma bileşenleri arasındaki girdi çıktı uyumsuzlukları, verilerdeki birim farklılıkları gibi eksiklik veya yanlışlıklar erken safhalarda tespit edilebilmektedir.

Yoğun miktarda işaret ve veri işleme yapılan algoritmalardan oluşan sistemlerde, sistemin çalıştığı ortamın simüle edilmesi ve algoritma testlerinin o ortamda yapılması çok önemlidir. Ancak her zaman sistemin bulunduğu ortamda olunamayacağı için o ortamdan gelen verilerle algoritmaların test edilmesi gerekmektedir. SAMSA aracıyla gerçek ortam verisi girdi olarak alınabilmekte ve sistem modeli bu verilerle çalıştırılabilmektedir.
Ayrıca SAMSA yazılımına ortam verisine uygun veri üretebilen bileşenler eklenmiştir. Bu bileşenler sayesinde üretilen verilerle gerçek çalışma ortamı simüle edilebilmekte ve oluşturulan sistem modeli ve algoritma bileşenleri gerçekçi verilerle çalıştırılmış olmaktadır.

3.3 Analiz

Algoritma bileşenlerinin olgunlaştırılabilmesi için analiz yapılması çok önemlidir. Geliştirilen bir algoritma bileşeninin gerçek ortam verileri ile çalıştırılması ve sonucunun analiz edilmesi ile ön görülemeyen bazı eksiklikler tespit edilebilmektedir. Böylece algoritmanın eksiklikleri, analiz sonuçlarından gelen geri beslemelerle kapatılmaktadır. SAMSA aracı, algoritmaları tek tek veya bir birleri ile etkileşimleri tanımlanmış bir sistem modelindeki durumlarıyla test yapma ve sonucu analiz etme imkanı sunmaktadır.
SAMSA aracıyla tanımlanmış bir sistem modelindeki algoritmaların çıktıları *Veri Editörü* bileşeniyle ayrı ayrı gözlemlenebilmektedir. Böylece sistem seviyesinde gözlemlenen bir hatanın sistem içerisinde gerçek yerinin bulunması kolaylaştırılmış olmaktadır.
SAMSA aracıyla *Karşılaştırma* bileşeni kullanılarak aynı işi yapan iki algoritma veya aynı algoritmanın iki versiyonu arasında karşılaştırma yapılabilmektedir. Bu yetenekle uygun algoritmanın veya versiyonun tespitinin yapılması mümkün kılınmaktadır.
Matlab yazılımı birçok mühendislik işlemleri için ve farklı türde grafik çizme açısından güçlü bir altyapıya sahiptir. Bu altyapı kullanılarak geliştirilmiş bileşenler SAMSA yazılımına kolayca dahil edilebilmekte ve bu bileşenler ile analiz işlemi kolaylaştırılmaktadır. Şekil 6'da örnek bir sistem modeli analizi görünmektedir.

Şekil 6. Örnek Analiz Ekranı

3.4 Farklı Kullanım Senaryoları

SAMSA yazılımı buraya kadar anlatılan yetenekleri farklı kullanım senaryoları ile desteklemektedir.
Sistem modeli daha önceden kaydedilmiş test verileri ile çalıştırılarak geçmişte yapılmış bir çalışma tekrarlanabilmektedir. Böylece yapılan değişikliklerin sistem üzerindeki etkisi fark edilebilmektedir. Aynı veri farklı sistem modellerinde çalıştırılarak değerlendirme yapılabilmektedir.
Farklı bir kullanım senaryosu da gerçek sistemde gerçek zamanlı olarak üretilen verilerin TCP/IP veya benzeri bir haberleşme protokolü ile alınması ve SAMSA'da tanımlanan sistem modelinin o verilerle çalıştırılması şeklindedir. Bu kullanım şeklinde algoritmaların geliştirildiği ortamların gerçek zaman uyumlu olmaması sebebiyle gerçek zamanlı performans elde edilememektedir. Ancak Sistemin genel çalışması hakkında bilgi vermektedir.
Bir başka çalışma senaryosunda SAMSA'da veri üretmek için oluşturulmuş bileşenler model içerisine dahil edilerek farklı farklı veriler üretilerek testler yapılabilmektedir. Bu bileşenlerin yeteneklerine göre gerçek ortam verisine yakın veriler üretilebilir ve sistemin daha tasarım aşamasında performansıyla alakalı fikir oluşturulabilir.
SAMSA ile algoritmaların tek tek veya birlikte çalışmalarının testi mümkündür. Beklenen çıktının ve üretilen çıktının karşılaştırma bileşenine girdi olarak verilmesi ile doğrulama amaçlı da kullanılabilmektedir.

4 Sonuç

SAMSA, girdisi ve çıktısı tanımlanmış algoritma bileşenleri kullanılarak ve bu algoritma bileşenlerinin arasındaki etkileşim tanımlanarak bir sistem modeli oluşturmaya, oluşturulan modelin çalıştırılması ile gerçek ortamı simüle etmeye ve çıkan sonuçların analizini yapmaya olanak sağlayan bir araçtır.
SAMSA aracı, modelleme altyapısı ile sistem tasarımının anlaşılabilirliğini artırmakta; simülasyon altyapısı ile algoritmaların gerçek ortamda nasıl davranacağını belirlemekte; analiz altyapısı ile de sistem performansının değerlendirilerek geri beslemeler yapılmasına imkan sağlamaktadır.
SAMSA aracı, her ne kadar Radar Elektronik Harp algoritmaları için tasarlanmaya başlanmış olsa da sahip olduğu altyapı sayesinde alandan bağımsız çalışabilmektedir.
SAMSA aracı, geliştirilmekte olan projelerde kullanılmakta ve ortaya çıkan ihtiyaçlar doğrultusunda aracın geliştirilmesine devam edilmektedir.

5 Kaynakça

1. Anu Maria, Introduction To Modeling And Simulation, Proceedings of the 1997 Winter Simulation Conference
2. http://www.mathworks.com/products/javabuilder/
3. http://docs.oracle.com/javase/6/docs/technotes/guides/jni/
4. http://docs.oracle.com/javase/tutorial/reflect/