İSTİNYE ÜNİVERSİTESİ

MÜHENDİSLİK VE DOĞA BİLİMLERİ FAKÜLTESİ

YAZILIM MÜHENDİSLİĞİ BÖLÜMÜ

YAZ492: BİTİRME TEZİ 2

Mayıs 2022

PROJE BAŞLIĞI

Paralel Mimarilerde Seyrek Alt Üçgen Matris Çözümü için
Graf Parçalaması ile Yük Dengelemesi

PROJE YAZARI

Abdülkadir Furkan YILDIZ -190701145

DANIŞMAN

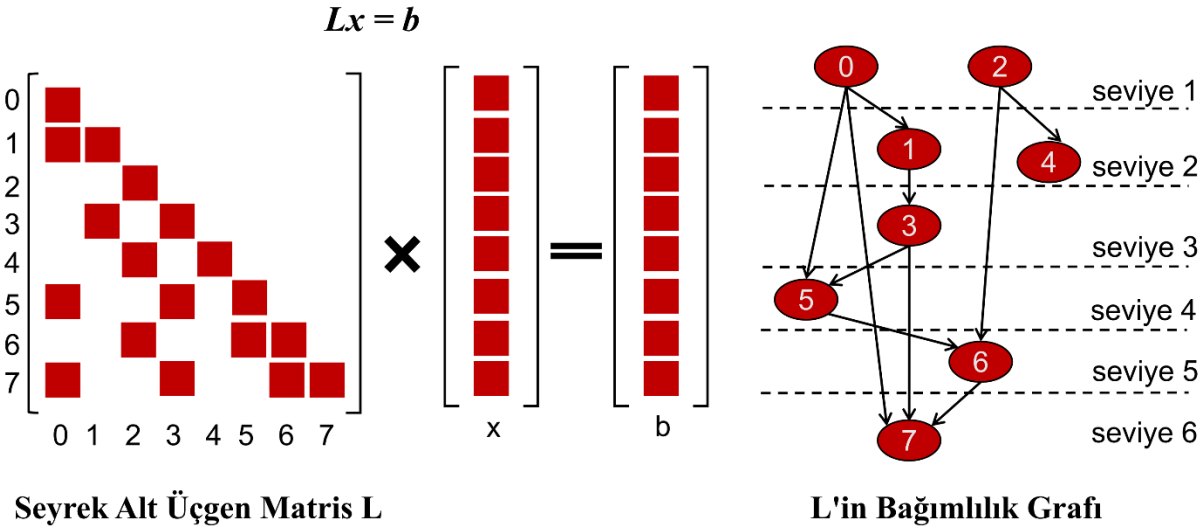
Dr. Öğr. Üyesi Buse YILMAZ

ÖZET

Chainbreaker paralel mimarilerde Seyrek Alt Üçgen Çözümü'nü optimize eden bir çerçevedir. Çerçevenin önemli bir modül yapılacak bağımlılık grafı dönüşümleri için stratejiler bütünü ve stratejilerin uygulanacağı seviyeleri belirleyen strateji koleksiyonu ve strateji seçme modülüdür. Projenin amacı, grafı aşağıdaki hedeflere ulaşılacağı şekilde dönüştürerek, seviyeler arasında ve içinde yük dengelemesini sağlayacak stratejileri ve uygulanacakları seviyeleri belirlemektir. Bu çalışmada, bir seyrek alt üçgen matrisin sınırlı paralellik gösteren parçalarının paralellik derecesini artırmak için bağımlılık grafını dönüştüren Chainbreaker çerçevesi için bir strateji koleksiyonu geliştirilecektir.

- Düşük paralellik derecesine sahip seyrek alt üçgen matris parçalarının paralelliğini graf dönüşümü ile artırarak, seyreklik yapısını daha homojen hale getirmek.

- Senkronizasyon noktalarına olan ihtiyacı azaltmak

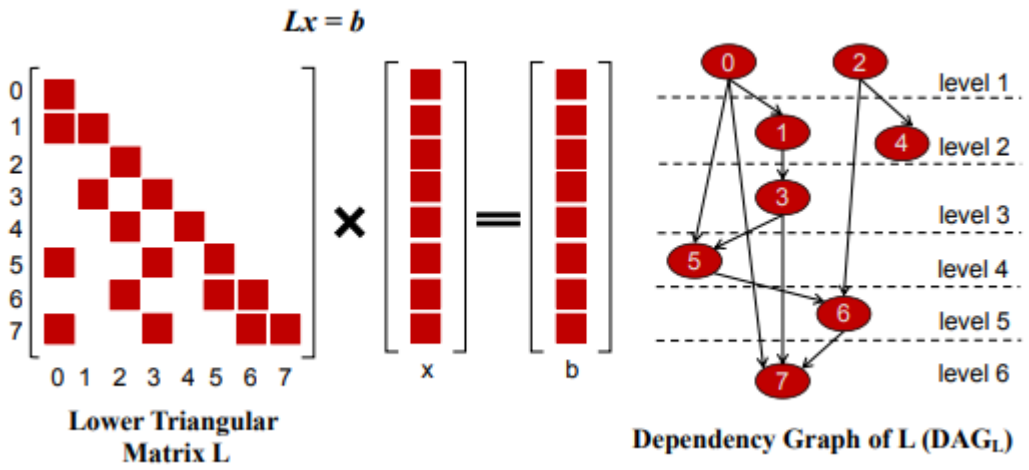


Şekil 1. Seyrek Alt Üçgen Matrislerin Lx = b

SUMMARY

Chainbreaker is a framework that optimizes the Sparse Lower Triangle Solution in parallel architectures. An important aspect of the framework is the set of strategies fort he milestones and the evaluation options and strategy selection to which the modüle will be applied. The aim of achieving the goal will be a Project whose graphic will be used. This review will develop a strategy fort he Chainbreaker framework, whose graph has not been completed to scale up the parallelism of a thin-screen mapped parallelism representation.

- To make the sparsity structure more homogeneous by increasing the parallelism of the sparse sub-triangular matrix parts with low degree of parallelism by graph transformation.
- Reducing the need for synchronization points.



Şekil 1. Seyrek Alt Üçgen Matris Çözümü işlemi Lx =b * L bağımlılık grafı

İÇİNDEKİLER

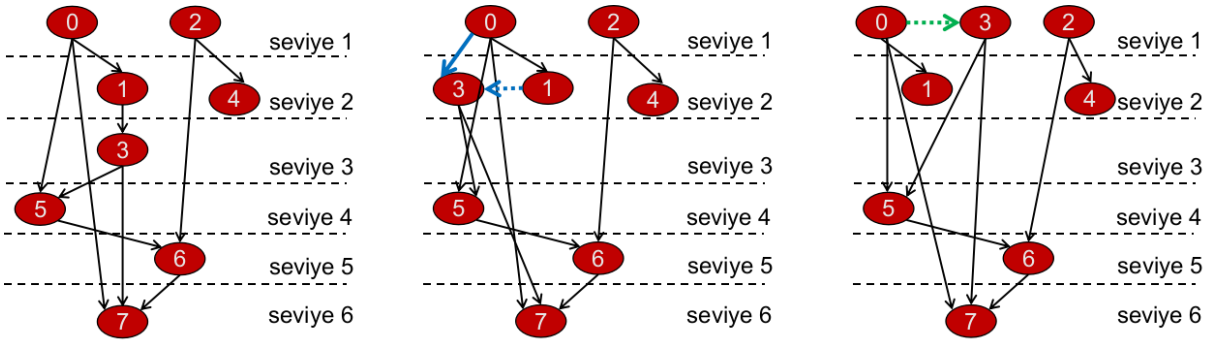
TABLO LİSTESİ (varsa)

FİGÜR LİSTESİ (varsa)

GİRİŞ

Bu çalışmanın amacı Seyrek Alt Üçgen Matrislerinin seyrek olan seviyelerini daha homojen bir hale getirmektir. Bundan dolayı matrislerin bağımlılık graflarında düğümlerin bağımlılıklarını kırarak seviye azaltma yöntemi ile bu homojenliğin sağlanması amaçlanmaktadır.

Bu çalışma bağımlılık grafı üzerinden Chainbreaker'ın yeniden yazma metodunu kullanarak, matrisin paralellik derecesinin az olduğu seviyelerdeki düğümlerin bağımlılıklarını kırarak bu seviyelerin paralellik derecesinin arttırılmasını hedeflemektedir. Bu paralelliği sağlarken düğümün kalmadığı seviyeleri ortadan kaldırarak matriside küçültmek amaçlanmaktadır.



a) Seyrek Alt Üçgen L'nin Bağımlılık Grafı

b) Satır 3'ün satır 1'e olan bağımlılığı kırıldı, satır 3 seviye 3'ten seviye 2'ye çıkarıldı

c) Satır 3'ün satır 0'e olan bağımlılığı kırıldı, satır 3 seviye 2'ten seviye 1'ye çıkarıldı

Satır denklemleri: x[0] = **b[0] / L[0][0],** x[1] = **(b[1] - L[1][0] * x[0]) / L[1][1]**
x[3] = (b[3] - L[3][1] * x[1]) / L[3][3]

Yeniden yazma işlemi:

b) x[3] = (b[3] - L[3][1] * (**(b[1] - L[1][0] * x[0]) / L[1][1]**)) / L[3][3]
c) x[3] = (b[3] - L[3][1] * (**(b[1] - L[1][0] * (b[0] / L[0][0])) / L[1][1]**)) / L[3][3]

Şekil 2. Yeniden Yazma Yöntemi

Bu çalışma, yukarıda gösterilen yeniden yazma işlemini matrislerin seyreklik yapısını homojen hale getirmek için bir strateji koleksiyonu oluşturmayı amaçlamaktadır. Bu koleksiyon ile beraber matrisler üzerinde ki paralelliği ve yük dengelemesinin

kolaylaştırılması amaçlanmaktadır. Matrislerin hesaplanmasında işlemlerin bölümlenmesini kolaylaştırarak daha hızlı bir yapı oluşturulması amaçlanmaktadır.

LİTERATÜR TARAMASI

Seyrek Alt Üçgen Matris Çözümü'nü optimize etme yaklaşımları açısından zengindir. Seviye kümesi oluşturma yöntemleri [1,2,3],birbirlerinden bağımsız olan düğümleri birbirleriyle gruplandırır. [3]'te, CPU'lar için seviye kümesi oluşturma yöntemine dayalı bir algoritma sunulmuştur.

Seviye kümesi oluşturma yöntemi [4]'de ve [5]'de NVIDIA GPU'lar için kullanılmıştır. Son zamanlarda yapılan bir çalışma [6], Sunway mimarisi için bu yaklaşımı, Seyrek Seviye Döşemesi adlı, verileri düzenleyen bir veri düzenleme önerisinde bulundu.

Kendi kendine zamanlama yöntemleri, seviye kümesi oluşturma yöntemine alternatif olarak, seviyelerin bariyer ihtiyaçlarını azaltmak için piyasaya çıktı. Bu yaklaşımın ilk örnekleri CPU'lar içindi [7]. Daha sonra, GPU'lar üzerinde çeşitli örnekler verildi [8,9,10,11]. [9]'da seviyeleri belirlemek için bir paralel bir sıralama algoritmasına ek olarak, düğümlerin her birini kendi satır bağımlılıklarını beklediği bir zamanlı strateji kullanıldı. [10]'da, bu yönteme benzer olarak zamanlama stratejisinin kullanıldığı ve seviyelerin bağımlılık derecelerinin üzerinde duran bir ön işleme aşaması önerilmektedir.[12]'de hem seviye kümesi oluşturma hem de kendi kendine zamanlama yöntemlerini kullanan hibrit bir yaklaşım CPU'lar için geliştirilmiştir.

Seyrek Alt Üçgen Matris Çözümü'nü optimal bir çözüm haline getirmek için bir diğer araştırma ise derleyiciler ile alakalıdır.[13]'de matrisin seyrek olan seviyelerini inceleyerek bu seviyeler için özel yeniden yazma işlemi önerilmiştir. Sympiler [14], derleme aşamasında sembolik bir analiz yapar ve seyrek seviyeler için bir yeniden yazma işlemi önerir ve bu önerilen işlem seviye kümesi oluşturma yöntemini kullanır.

METOT, UYGULAMA ve TESTLER

Metot

Bir önceki raporda belirtilen ve BAŞARIM 2022 konferansı için bildiri olarak gönderilen ve kabul alan bildirideki stratejimizin üzerine geliştirmeler yapılarak yeni bir strateji geliştirilmiştir. Bir önceki stratejimiz genel olarak avgLevelCost'a bakarak yeniden yazma işlemini yapmaktada idi. Bu stratejinin eksik yönlerinin olmasından dolayı yeni bir strateji geliştirdik. Bu eksiklikler genel olarak düğümlerin girdi düğümlerinin fazla olmasından dolayı yüksek seviye maliyeti ve yüksek seviye maliyetine sahip seviyelere çok fazla düğümün yeniden yazılması gibi sorunlar yüzünden paralelliği artırma noktasında eksik kaldı.

Paralelliği artırmak için bağımlılık grafiğini dönüştürmek çok maliyetli bir iştir, çünkü çok sayıda satırı yeniden yazıldığında maliyetler katlanarak artar. Bundan dolayı belirli bir metot çerçevesinde bu işlem yapıldığında bu maliyet belirli bir şekilde artar ve paralellik bu derece artmış olur. Biz burada 3 kriterli sınırlamaları olan bir streteji sunuyoruz. Bu 3 kriter avgCostPerLevel, avgNumRowsPerLevel ve avgIndegreePerLevel olarak belirlendi.

Bu kriterlerin genel amacı seviyelerin genel maliyetini azaltmaktır. Öncelikli olarak her matris için bu kriterleri hesaplatmamız gerekiyor. avgCostPerLevel, avgNumRowsPerLevel ve avgIndegreePerLevel değerleri hesaplanır:

- avgCostPerLevel: Ortalama Seviye Maliyeti
- avgNumRowsPerLevel : Seviyelerde ki ortalama düğüm sayısı
- avgIndegreePerLevel : Düğümlerin giren düğüm değerlerinin ortalaması.

Bu iki metodu kullanarak uygulamamızda kullanacağımız kriterler hesaplanır ve gerekli değişkenlere aktarılarak yeniden yazma işlemini yaparken sürekli olarak hesaplanmaz ve tüm matrisin yeniden yazma işlemi bitene kadar sabit kalır.

Uygulama

Şekil 2, satırların yeniden yazılmasına dayalı graf dönüştürme yaklaşımnı göstermektedir. Bizim önerdiğimiz stratejide verilen matris CSR formatında bir matrise çevirilir ve yukarıdaki şekilde ki gibi yeniden yazma işlemi yapılır ancak burada yeniden yazma işleminin yapılabilmesi için üç adet kriterin sağlanması gerekmektedir. Bu örnekte seviye 3'e 2 kere yeniden yazma işlemi uygulanmıştır. Burada seviye 2 hedef seviyedir.

Eğer kriterlerden herhangi biri sağlanmaz ise hedef seviye bir sonraki seviye olmaktadır. Bu şekilde tüm seviyeler gezilir ve yeniden yazma işlemi yapılır. Yeniden yazma işlemi tamamlandıktan sonra yapılan tüm işlemler bir rapor halinde matrisin ID'sine sahip bir dosya oluşturulur.

Testler ve Sonuçlar

Önerilen stratejinin kullanımı için 9 adet matris üzerinden test işlemi gerçekleştirildi. Bu matrislerin özellikleri Tablo 1 ‘de verilmiştir. Bu matrislerin test ve sonuç verileri Sonuç kısmında detaylı şekilde verilmiştir.

Matris İsmi	Matrisin Türü	0’dan Farklı Girdi Sayısı	Satır Sayısı
Lung2	Hesaplamalı Akışkanlar Dinamiği Problemi	273.647	109.460
Torso2	2D/3D Problemi	574.718	115.967
Venkat01	Hesaplamalı Akışkanlar Dinamiği Problemi	890.108	62.424
Bcsstk17	Yapısal Problem	219.812	10.974
Si41Ge41H72	Teorik / Kuantum Kimyası Problemi	7.598.452	185.639
Bcsstk37	Yapısal Problem	583.240	25.503
Pwtk	Yapısal Problem	5.926.171	217.918
Cfd2	Hesaplamalı Akışkanlar Dinamiği Problemi	2.605.669	123.440
Gearbox	Yapısal Problem	4.617.075	153.746

Tablo1 . Matrislerin Özellikleri

SONUÇ

Bir önceki kısımda önerilen strateji üzerinden Tablo 1.'de verilen matrisler ile yapılan testlerin sonuçları ve grafikleri Tablo 2 ve Tablo 3' de verilmiştir.

Bu çalışmada Seyrek Alt Üçgen Matris Çözümü'nün yeni ve özgün bir yöntem olan, Üç kriterli yeniden yazma yöntemini öneriyoruz. Seyrek bir alt üçgen matrisin paralellik derecesi az olan parçalarını bu yöntemle dönüştürerek bu parçaların paralellik derecesini arttırmayı amaçlıyoruz.

Yaptığımız testler sonucunda önerdiğimiz stratejinin güçlü yanlarını ve limitlerini analiz ettik bunlar şunlardır:

Güçlü Yönleri:

- FLOPS(kayan noktasal hesaplamalar) değerleri BAŞARIM konferansında önerilen stratejiye göre çok yüksek seviyelere çıkmadı ve belirli bir oranda sabit kaldı.
- Seviyelerdeki düğüm sayılarında bir yığılma olmadı, bu precision sorunlarına yol açmıştı ve bunun önüne geçtik.

Limitleri:

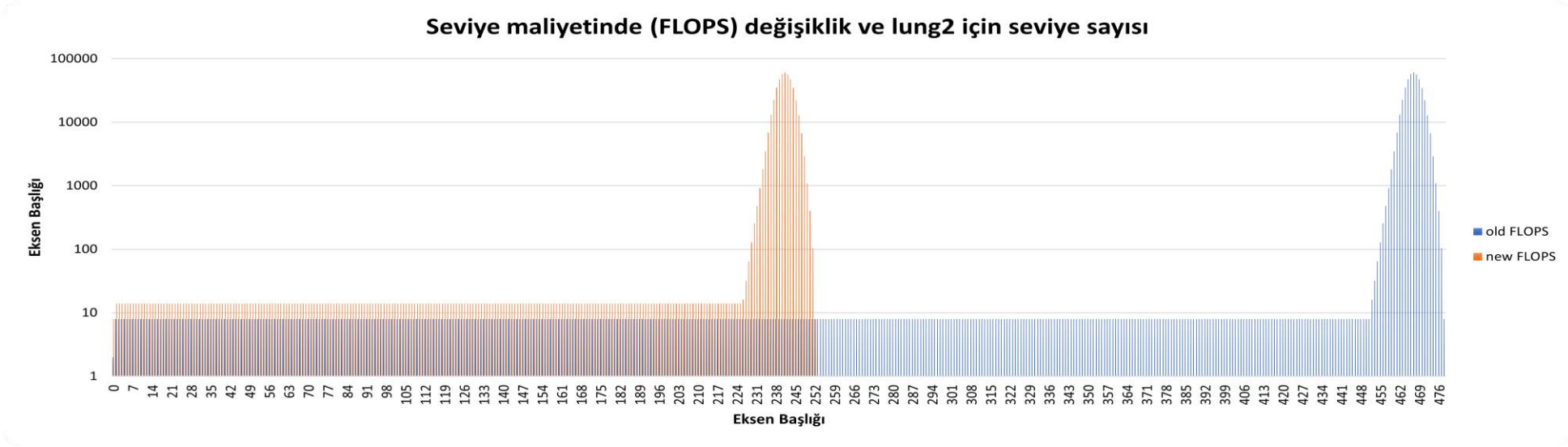
- Yeniden yazılan düğümün girdi düğümünün fazla olduğu matrislerde FLOPS değerleri ortalamanın altında kalmasına ragmen yeniden yazma işlemi yapılmadı.
- Bu işlemin yapılmamasından dolayı seviyelerdeki düğüm sayıları sabit kaldı ve yeniden yazma işlemi yapılması gereken seviyelere yapılamadığı için sabit kaldı ve yeniden yazılan düğüm sayısı çok az oldu.
- Yeniden yazılan düğümün girdi düğümünün fazla olduğu satırlarda, hedef seviyenin toplam maliyeti ortalama seviye maliyetinin altında kalmış olmasına ragmen yeniden yazma işlemi yapılmadı. Bu da seviye sayısının yeterince azaltamamıza sebep oldu ve seviyeler arasındaki homojenlik istenilen seviyeye çıkmadı.

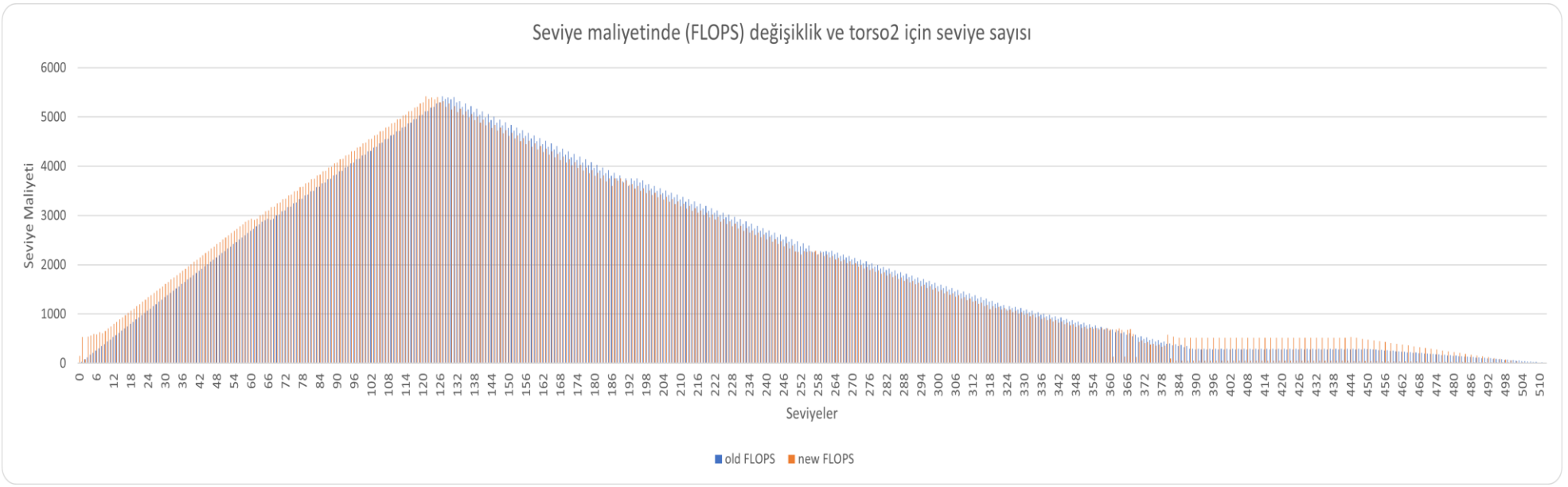
Test Veri Sonuçları:

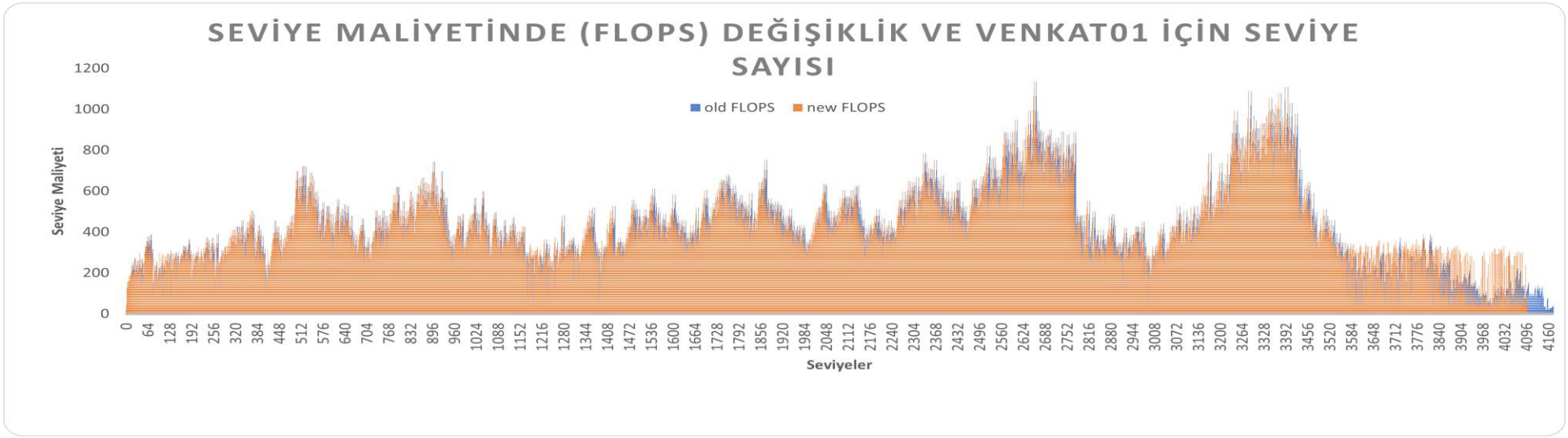
Lung2	Yeniden Yazmadan Önce	Yeniden Yazmadan Sonra	Varyans
Yeniden Yazmadan sonraki FLOPS Varyansı	37762556,04	700088328,51(%1753.92)	
Seviye Sayısı	479	253(%47 -)	0,4718
Toplam Seviye Maliyeti	437834	437382(%0.103 -)	1,03
Ortalama Seviye Maliyeti	914	1728(%89)	0,89
Yeniden Yazılan Satır Sayısı		452	
Torso2	Yeniden Yazmadan Önce	Yeniden Yazmadan Sonra	Varyans
Yeniden Yazmadan sonraki FLOPS Varyansı	2750547,658	2748445,202(0.076 -)	
Seviye Sayısı	513	507(%1.17 -)	0,0116
Toplam Seviye Maliyeti	103347	103319(0.027 -)	1,00
Ortalama Seviye Maliyeti	2014,56	2037,85(%1.156)	0,0115
Yeniden Yazılan Satır Sayısı		2370	
Venkat01	Yeniden Yazmadan Önce	Yeniden Yazmadan Sonra	Varyans
Yeniden Yazmadan sonraki FLOPS Varyansı	35139,82288	34343,6236(%2.266 -)	
Seviye Sayısı	4176	4100(%1.82 -)	0,0181
Toplam Seviye Maliyeti	171177	171733(%0.325)	2,68
Ortalama Seviye Maliyeti	411.346	418.861(%1.827)	0,0182
Yeniden Yazılan Satır Sayısı		2708	
Si41Ge41H72	Yeniden Yazmadan Önce	Yeniden Yazmadan Sonra	Varyans
Yeniden Yazmadan sonraki FLOPS Varyansı	21856,0429	31048,93309(%42)	
Seviye Sayısı	181783	105297(%42 -)	0,4207
Toplam Seviye Maliyeti	150113	149348(0,51 -)	5,10
Ortalama Seviye Maliyeti	82.577	141.835(%71)	0,7176
Yeniden Yazılan Satır Sayısı		77684	
Bcsstk17	Yeniden Yazmadan Önce	Yeniden Yazmadan Sonra	Varyans
Yeniden Yazmadan sonraki FLOPS Varyansı	44699,03721	43385,67736(%2.93 -)	
Seviye Sayısı	1332	1298(%2.55 -)	0,0255
Toplam Seviye Maliyeti	428650	428630(%0.005 -)	4,67
Ortalama Seviye Maliyeti	321.809	330.223(%2.61)	0,0261
Yeniden Yazılan Satır Sayısı		357	

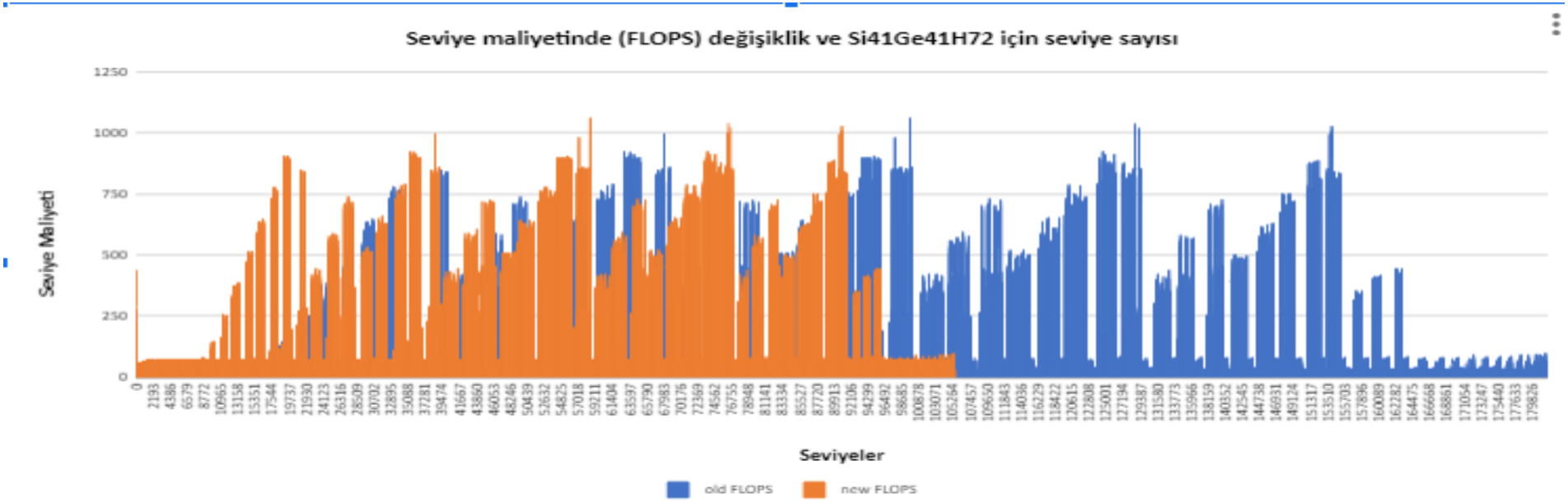
Bcsstk37	Yeniden Yazmadan Önce	Yeniden Yazmadan Sonra	Varyans
Yeniden Yazmadan sonraki FLOPS Varyansı	29581,57874	28771,62973(%2.7 -)	
Seviye Sayısı	5030	4730(%5.96 -)	0,0596
Toplam Seviye Maliyeti	114098	114060(%0.033 -)	4,56
Ortalama Seviye Maliyeti	226.834	241.112(%6.2)	0,0629
Yeniden Yazılan Satır Sayısı		2175	
pwtk	Yeniden Yazmadan Önce	Yeniden Yazmadan Sonra	Varyans
Yeniden Yazmadan sonraki FLOPS Varyansı	23726,79596	23784,31426(0.242)	
Seviye Sayısı	142168	141758(%0.288 -)	0,0028
Toplam Seviye Maliyeti	116344	116340(%0.003 -)	9,00
Ortalama Seviye Maliyeti	81.835	82.069(%0.286)	0,0028
Yeniden Yazılan Satır Sayısı		461	
Cfd2	Yeniden Yazmadan Önce	Yeniden Yazmadan Sonra	Varyans
Yeniden Yazmadan sonraki FLOPS Varyansı	165563,5139	159677,7788(%3.5 -)	
Seviye Sayısı	4357	4240(%2.68 -)	0,0268
Toplam Seviye Maliyeti	308790	308710(%0.026 -)	2,59
Ortalama Seviye Maliyeti	708.721	728.089(%2.73)	0,0273
Yeniden Yazılan Satır Sayısı		6115	
Gearbox	Yeniden Yazmadan Önce	Yeniden Yazmadan Sonra	Varyans
Yeniden Yazmadan sonraki FLOPS Varyansı	3011755,018	2986382,509(%0.84 -)	
Seviye Sayısı	4586	4436(%3.27 -)	0,0327
Toplam Seviye Maliyeti	908040	907691(%0.038 -)	-9,00
Ortalama Seviye Maliyeti	1980	2046(%3.33)	0,0333
Yeniden Yazılan Satır Sayısı		4042	

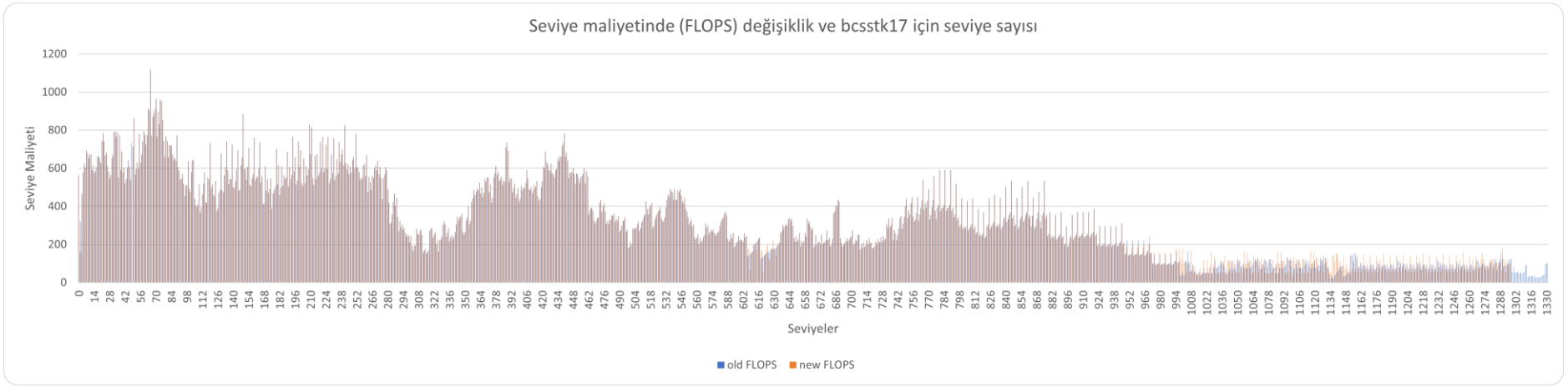
Tablo 2. Test Sonuçları

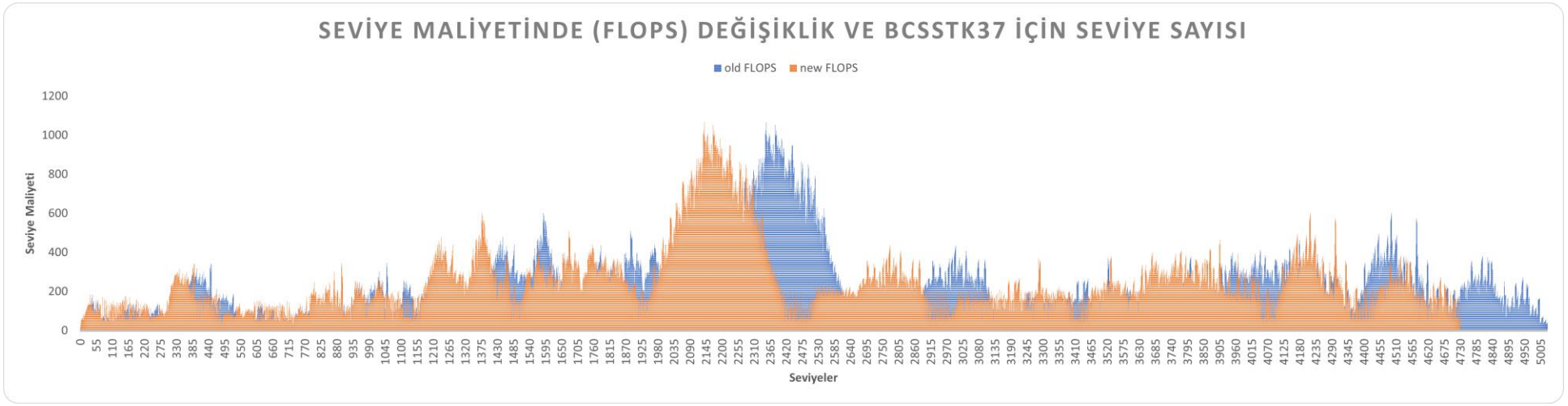

Seviye maliyetinde (FLOPS) değişiklik ve lung2 için seviye sayısı
Eksen Başlığı
100000
10000
1000
100
10
1
Eksen Başlığı
old FLOPS
new FLOPS

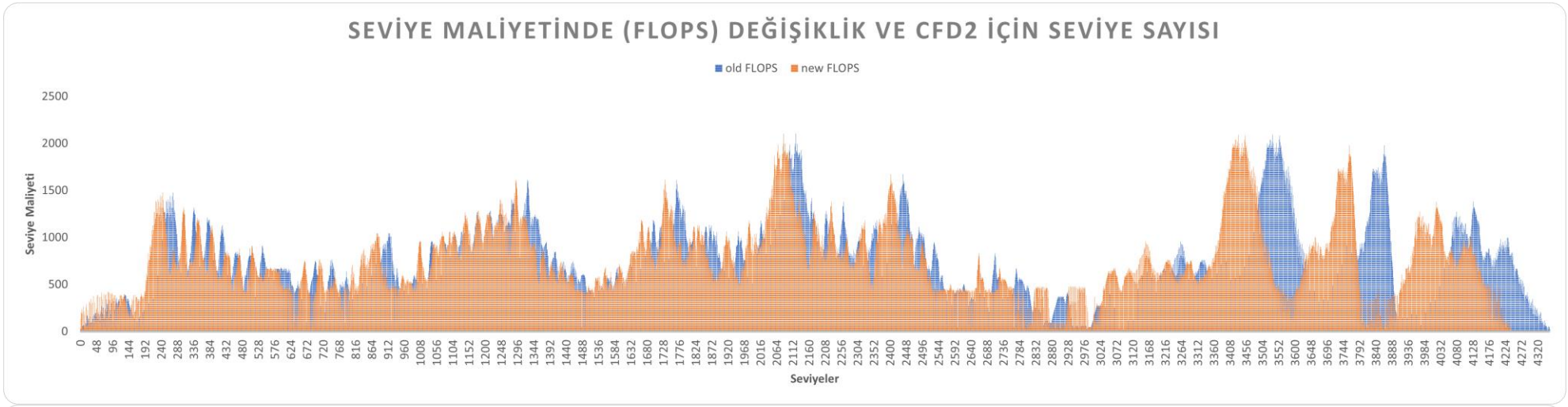

Seviye maliyetinde (FLOPS) değişiklik ve torso2 için seviye sayısı
Seviye Maliyeti
6000
5000
4000
3000
2000
1000
0
Seviyeler
old FLOPS
new FLOPS

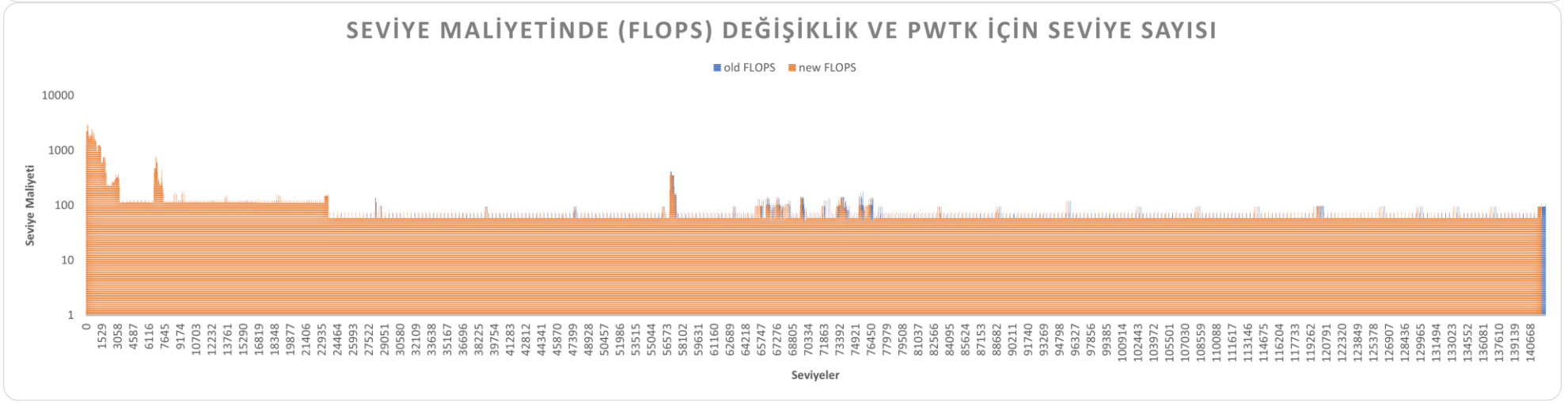

SEVİYE MALİYETİNDE (FLOPS) DEĞİŞİKLİK VE VENKAT01 İÇİN SEVİYE SAYISI
old FLOPS
new FLOPS
Seviye Maliyeti
Seviyeler

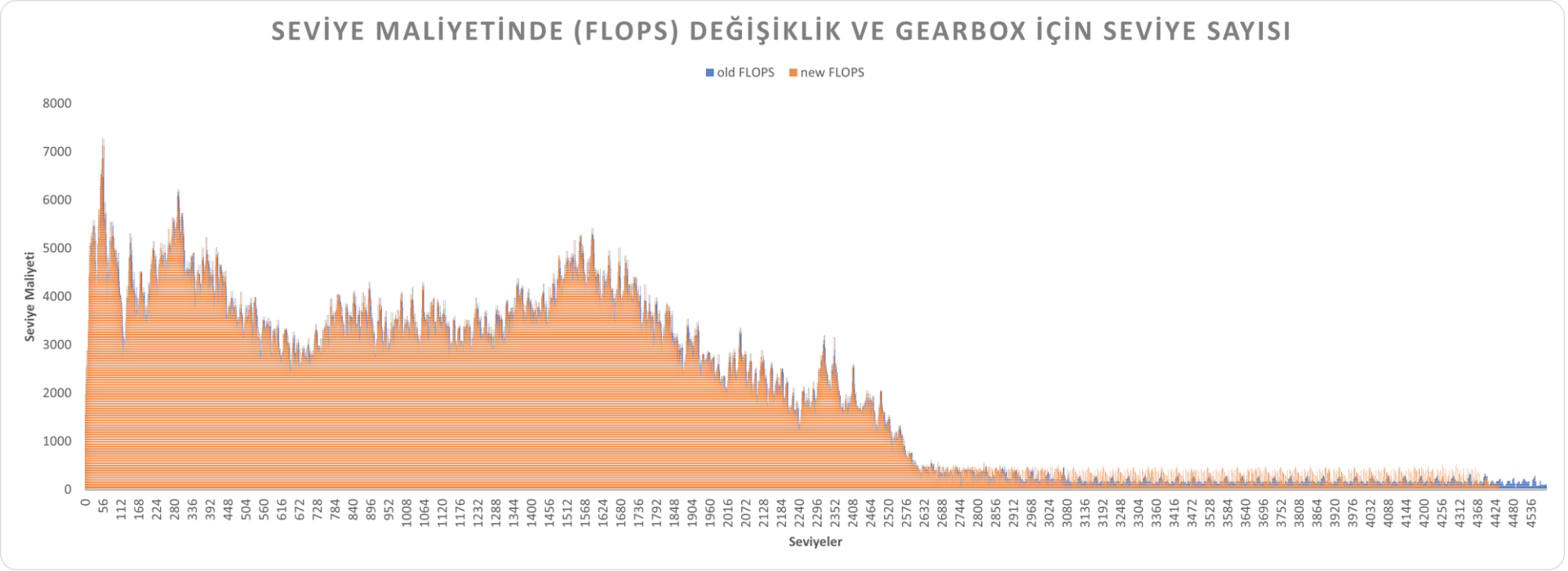

Seviye maliyetinde (FLOPS) değişiklik ve Si41Ge41H72 için seviye sayısı
Seviye Maliyeti
Seviyeler
old FLOPS
new FLOPS


Seviye maliyetinde (FLOPS) değişiklik ve bcsstk17 için seviye sayısı
Seviye Maliyeti
Seviyeler
old FLOPS
new FLOPS


SEVİYE MALİYETİNDE (FLOPS) DEĞİŞİKLİK VE BCSSTK37 İÇİN SEVİYE SAYISI
old FLOPS
new FLOPS
Seviye Maliyeti
Seviyeler


SEVİYE MALİYETİNDE (FLOPS) DEĞİŞİKLİK VE CFD2 İÇİN SEVİYE SAYISI
old FLOPS
new FLOPS
Seviye Maliyeti
Seviyeler


SEVİYE MALİYETİNDE (FLOPS) DEĞİŞİKLİK VE PWTK İÇİN SEVİYE SAYISI
old FLOPS
new FLOPS
Seviye Maliyeti
Seviyeler



Tablo 3. FLOPS Değerlerinin Grafik Üzerinde ki Görünümü

GELECEK ÇALIŞMALAR

Daha sonraki çalışmalarda Sonuç kısmında ele alınan yönleri iyileştirme ve onlar üzerinden önerilen matrisin seyreklik yapısını daha homojen hale getirmek için çalışmalar yapılacaktır.

KAYNAKÇA

[1] Edward Anderson ve Yousef Saad. “Solving Sparse Triangular Linear Systems on Parallel Computers”. International Journal of High Speed Computing 1, 1989.

[2] Joel H. Saltz. “Aggregation Methods for Solving Sparse Triangular Systems on Multiprocessors”. SIAM J. Sci. Stat. Comput. 11, 1. 1990.

[3] Edward Rothberg ve Anoop Gupta. “Parallel ICCG on a hierarchical memory multiprocessor — Addressing the triangular solve bottleneck”. Parallel Comput. 18,7. 1992.

[4] Maxim Naumov. “Parallel Solution of Sparse Triangular Linear Systems” in the Preconditioned Iterative Methods on the GPU. Technical Report, 2011.

[5] Ruipeng Li ve Yousef Saad. “GPU-accelerated preconditioned iterative linear solvers.” The Journal of Supercomputing 63, 2. 2013.

[6] Xinliang Wang, Weifeng Liu, Wei Xue, ve Li Wu. “swSpTRSV: A Fast Sparse Triangular Solve with Sparse Level Tile Layout on Sunway Architectures.” SIGPLAN Not. 53, 1. 2018.

[7] Steven W. Hammond ve Robert Schreiber. “Efficient ICCG on a Shared Memory Multiprocessor”. International Journal of High Speed Computing 04, 01 1992 .

[8] José I. Aliaga, Ernesto Dufrechou, Pablo Ezzatti, and Enrique S. Quintana-Ortí. “Accelerating the task/dataparallel version of ILUPACK’s BiCG in multi-CPU/GPU configurations”. Parallel Comput. 85, 79 – 87, 2019.

[9] Ruipeng Li. “On Parallel Solution Of Sparse Triangular Linear Systems in CUDA”. Technical Report. arXiv:1710.04985v1, 2017.

[10] Weifeng Liu, Ang Li, Jonathan Hogg, Iain S. Duff, ve Brian Vinter. “A Synchronization-Free Algorithm for Parallel Sparse Triangular Solves”. In Proceedings of the 22Nd International Conference on Euro-Par 2016: Parallel Processing- Volume 9833. Springer-Verlag New York, Inc., New York, NY, USA, 617–630, 2016.

[11] Weifeng Liu, Ang Li, Jonathan D. Hogg, Iain S. Duff, ve Brian Vinter. “Fast synchronization-free algorithms for parallel sparse triangular solves with multiple right-hand sides”. Concurrency and Computation: Practice and Experience 29, 21, 2017.

[12] Jongsoo Park, Mikhail Smelyanskiy, Narayanan Sundaram, ve Pradeep Dubey. “Sparsifying Synchronization for High-Performance Shared-Memory Sparse Triangular Solver.” In Proceedings of the 29th International Conference on Super-computing – Volume 8488 (ISC 2014). Springer-Verlag New York, Inc., New York, NY, USA, 124–140, 2014.

[13] H. Rong, J. Park, L. Xiang, T. A. Anderson, ve M. Smelyanskiy. “Sparso : Context-driven optimizations of sparse linear algebra.” In 2016 International Conference on Parallel Architecture and Compilation Techniques (PACT). 247–259, 2016.

[14] Kazem Cheshmi, Shoaib Kamil, Michelle Mills Strout, ve Maryam Mehri Dehnavi. “Sympiler: Transforming Sparse Matrix Codes by Decoupling Symbolic Analysis.” In Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (SC ’17). ACM, New York, USA, Article 13, 13 pages, 2017.

EKLER (varsa)

Not: Varsa kodlar, framework ve başka ek bilgiler bu bölümde danışman onayına yerleştirilebilir.

Not: Projenin son hali somut bir proje olduğu varsayımıyla kodlar, çalışır program vb. En son raporun danışmana fiziki tesliminde CD üzerinde yazılmalı ve final raporla birlikte danışman hocanıza Kurul sonrası verilmelidir.