Parallel Computing / Server Solutions

Mustafa Onur Özkan
MATLAB Information Systems Specialist

FİGES
applied physics & mathematics

MathWorks®
Authorized Reseller

Agenda

01 Introduction To The Presentation

02 MATLAB Parallel Computing / Server

03 Installation, Configuration and System Requirement

04 Usage Scenarios / User Story

05 Common Problems and Solutions

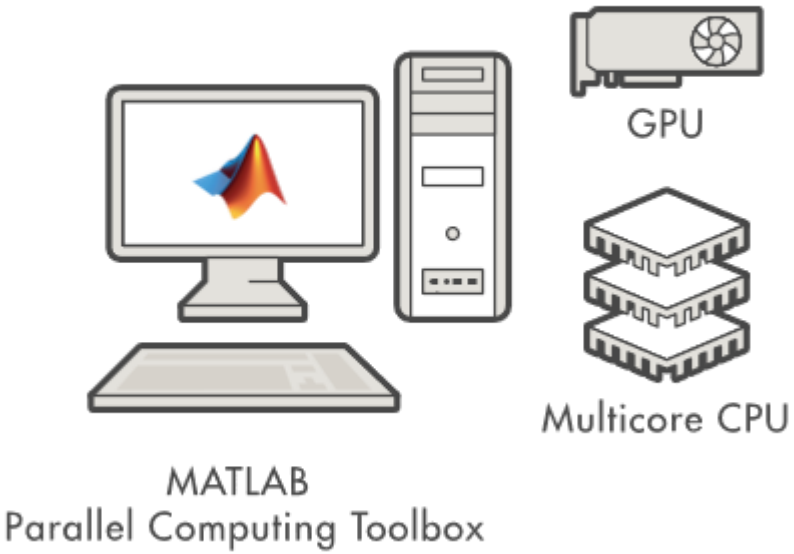
Purpose and Scope of the Presentation

- Purpose of the presentation: To examine the features, installation and use of MATLAB Parallel Server/ Computing Solution in detail.

- Scope: Parallel Computing Basic concepts, installation and configuration, usage scenarios, advanced topics, frequently encountered problems and solutions.

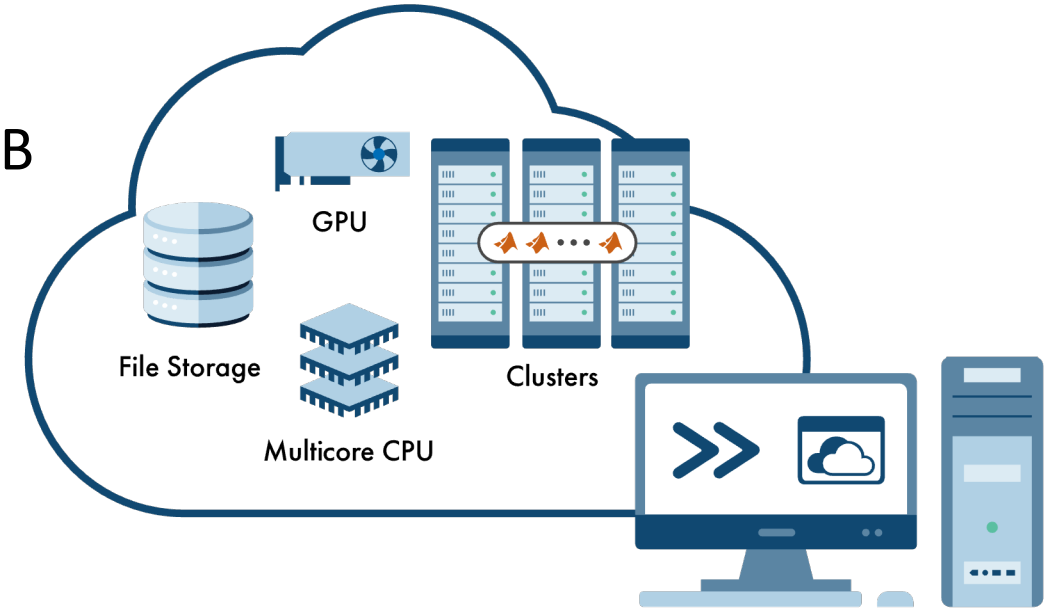
What is Parallel Computing Toolbox?

- Parallel Computing Toolbox offers MATLAB users the ability to perform parallel computing on their local machines.
- This Toolbox allows your local computer to speed up operations by using its processors and GPUs.



What is MATLAB Paralel Server ?

- MATLAB Parallel Server allows you to run MATLAB operations in parallel on a cluster of computers.
- It increases the speed of calculations by distributing the workload across multiple computers.



Why Do We Need Parallel Computing?

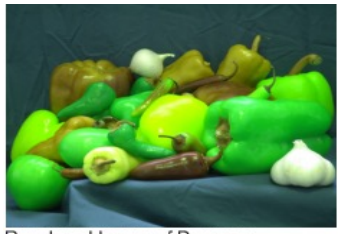
- Fast processing of large data sets.
- Accelerating simulation and modeling processes.
- Cost savings through effective use of resources.

Parallel - Enabled Toolboxes (MATLAB Product Family)

Image Processing

Batch Image Processor, Block Processing, GPU-enabled functions

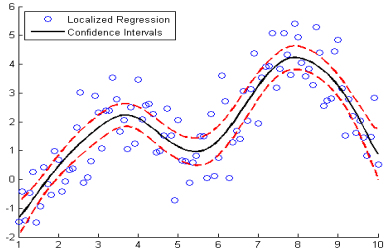
Original Image of Peppers



Recolored Image of Peppers

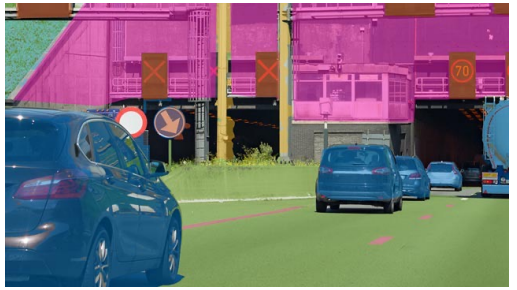
Statistics and Machine Learning

Resampling Methods, k-Means clustering, GPU-enabled functions



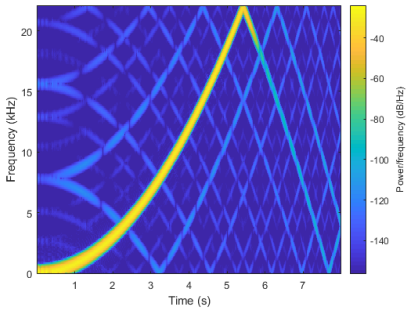
Deep Learning

Deep Learning, Neural Network training and simulation



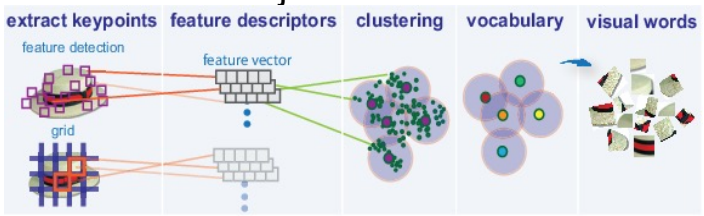
Signal Processing and Communications

GPU-enabled FFT filtering, cross correlation, BER simulations



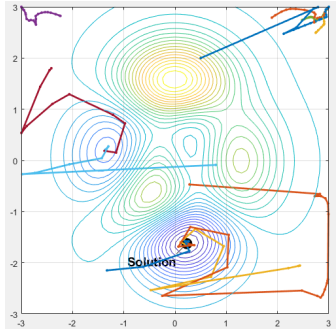
Computer Vision

Bag-of-words workflow, object detectors



Optimization & Global Optimization

Estimation of gradients, parallel search



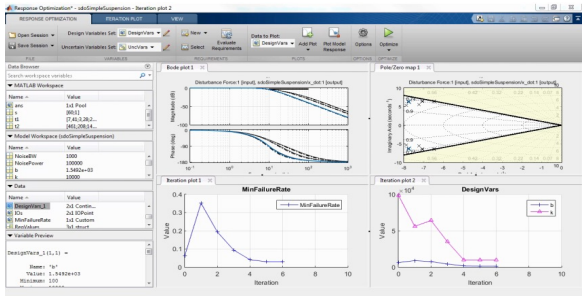
FİGES applied physics & mathematics | MathWorks Authorized Reseller

BİZİ TAKİP EDİN!

Parallel - Enabled toolboxes (Simulink Product Family)

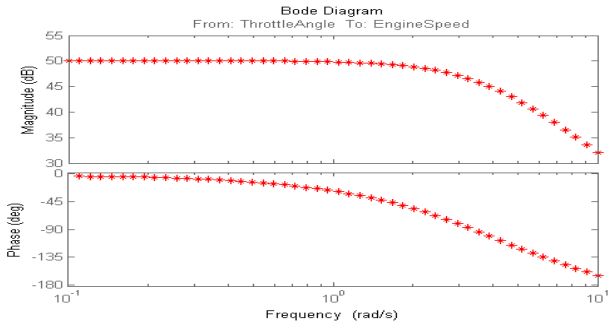
Simulink Design Optimization

Response optimization, sensitivity analysis, parameter estimation



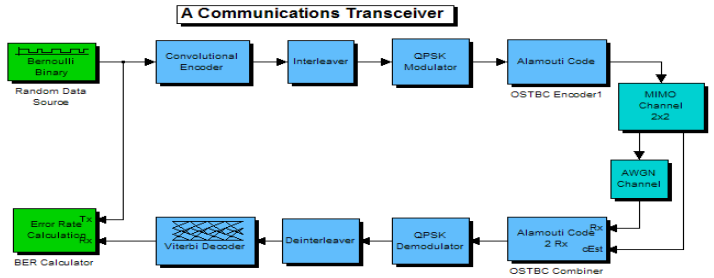
Simulink Control Design

Frequency response estimation



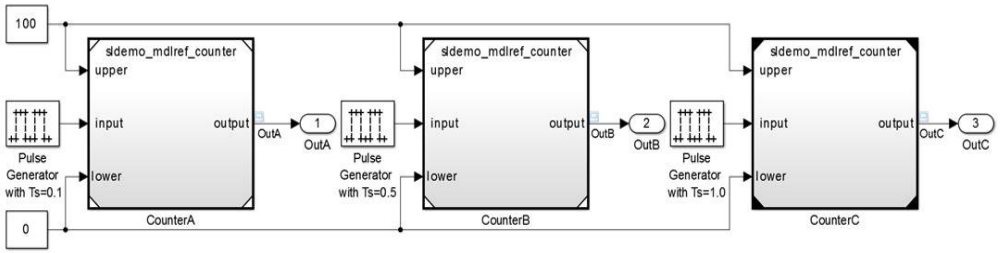
Communication Systems Toolbox

GPU-based System objects for Simulation Acceleration



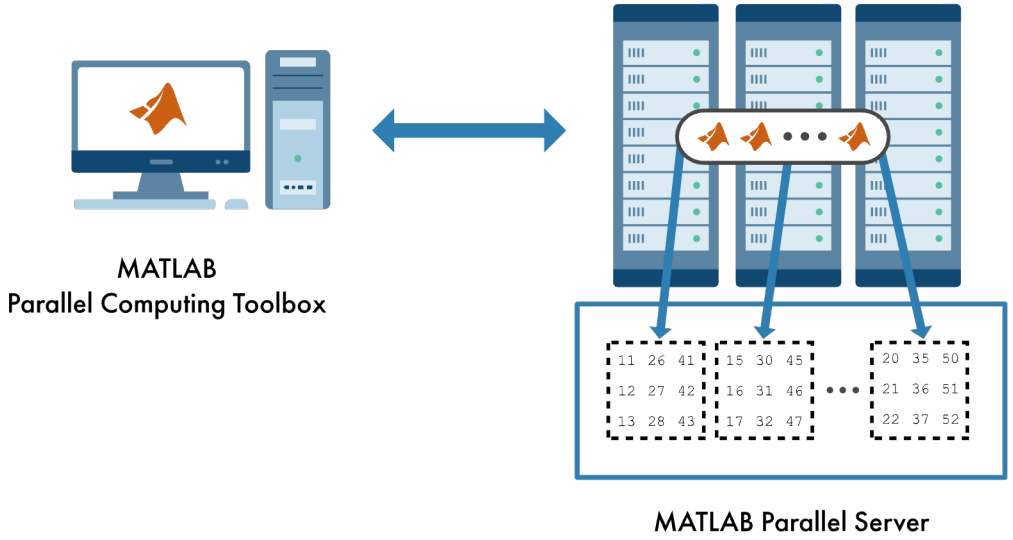
Simulink/Embedded Coder

Generating and building code



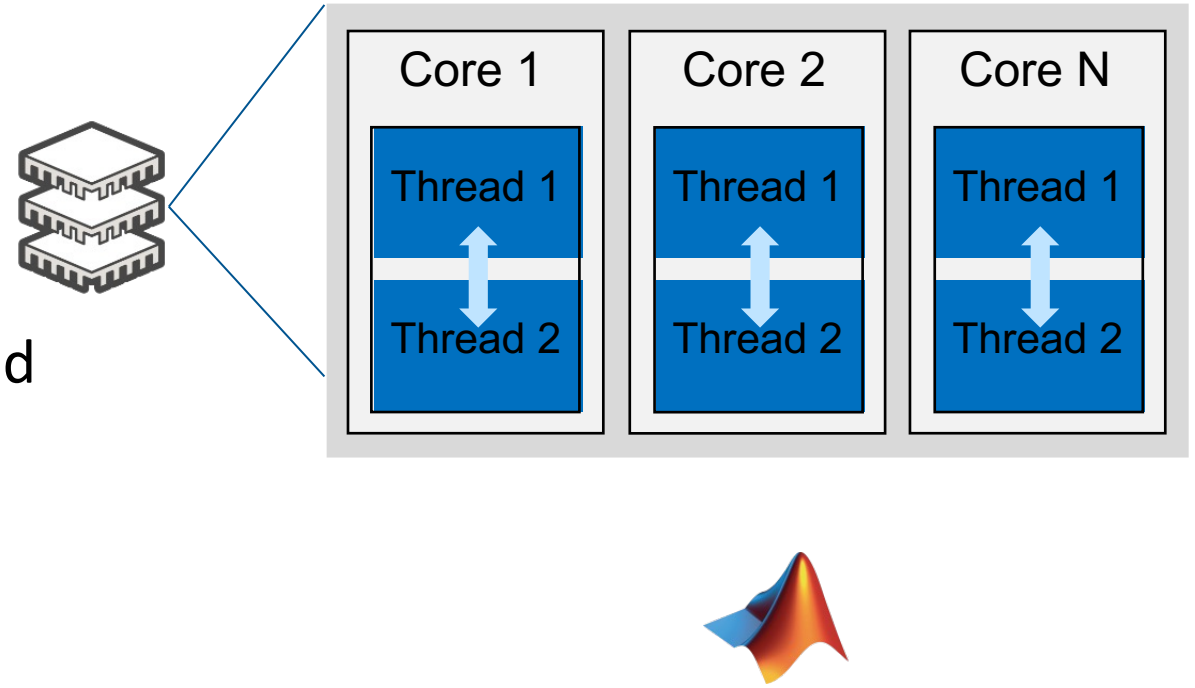
MATLAB Parallel Server Basic Concepts

- Cluster: The structure where computational resources come together.
- Worker: Independent processing units that perform parallel computation.
- Job: The set of operations or tasks to be performed.
- Task: Independent processing unit within a job.



MATLAB Parallel Server Architectural

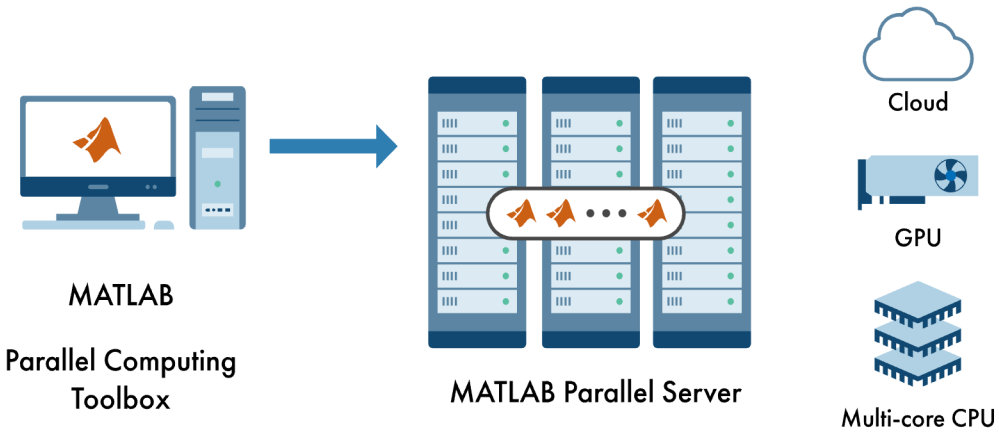
- Server-client structure.
- Scheduler and Resource Manager components.
- Communication between MATLAB client and worker.



MATLAB Parallel Server Working Principle

- Creation of jobs and tasks.
- Distribution and execution of tasks in parallel.
- Collection and processing of results.

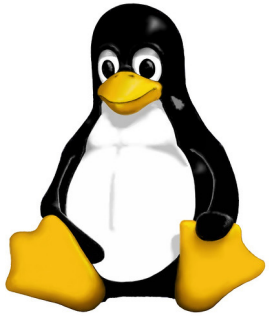
```
parpool("HPC",1000);

parfor ii = 1:3000
    c(ii,:) = eig(rand(1000));
end
```

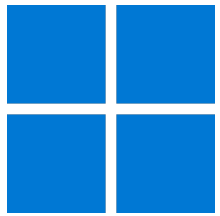


Installation and Configuration – System Requirement

System Requirement

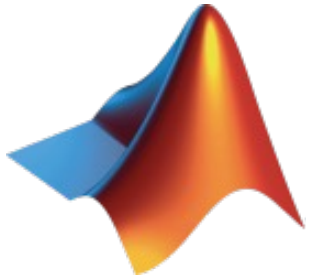
- Supported operating systems.
- Required hardware components (CPU, GPU, RAM, Disk, etc.)

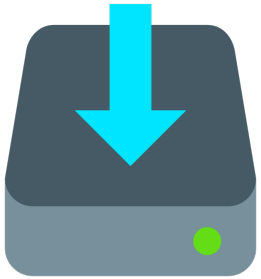





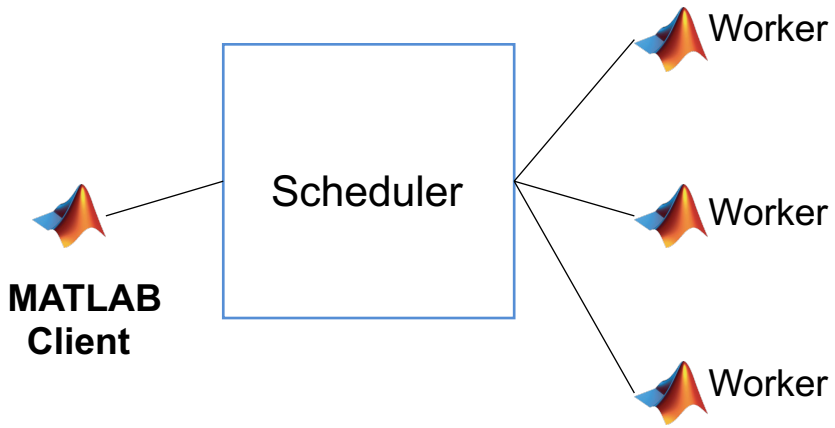
AMD intel NVIDIA Red Hat

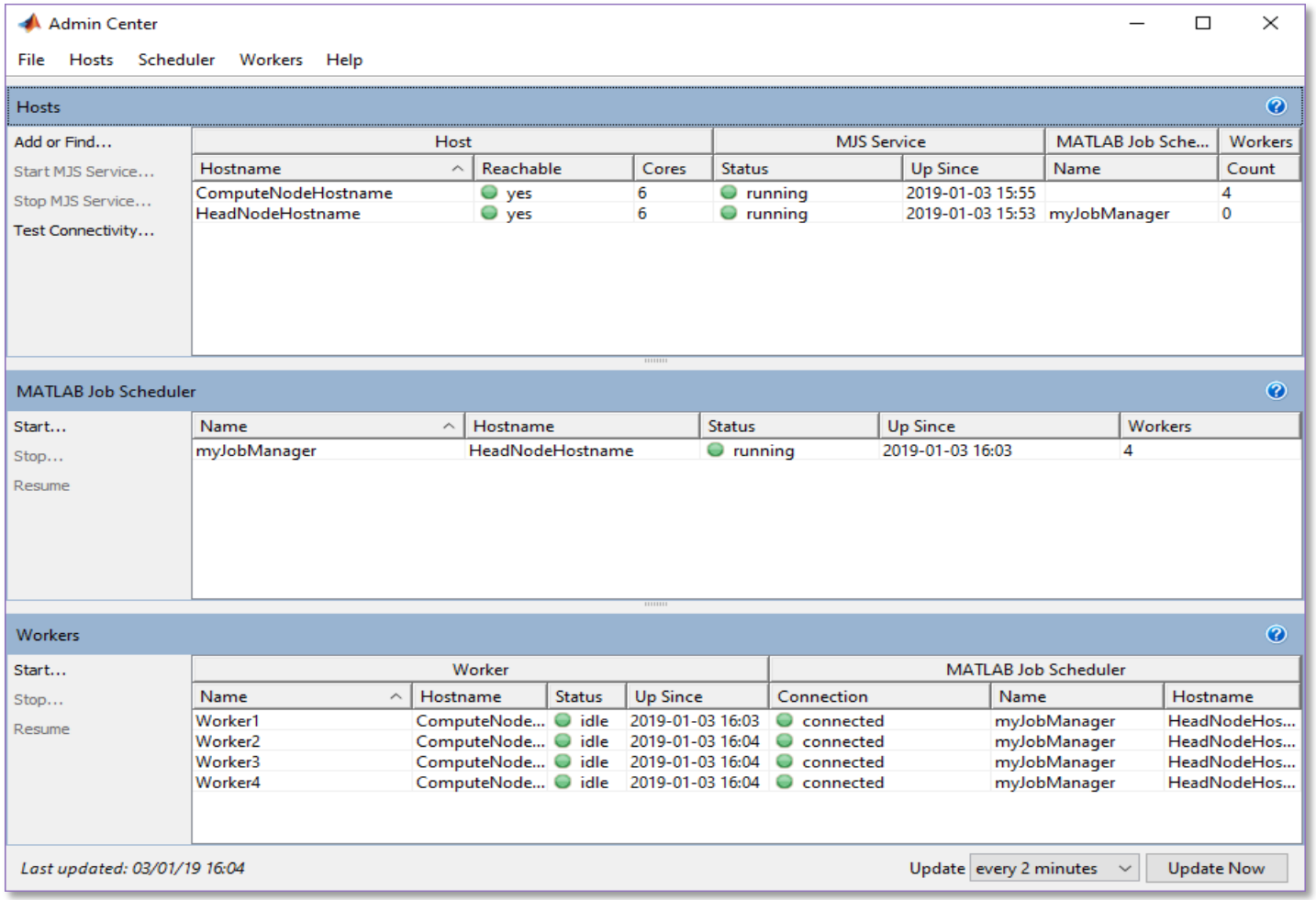
Supported Job Schedulers





MATLAB MJS





Parallel Loops / Parfor Loop

- Parfor loop is used to run independent iterations in parallel.
- Parfor runs the iteration independently and speeds up the process.
- In this example, the total for each row is calculated in parallel.

Editor - C:\Users\onur.ozkan\OneDrive - figes.com.tr\Masaüstü\Optimization Example\bigdata.m

myObjectiveFunction.m | mainNoneParallel.m | mainParallel.m | bigdata.m

```
1   % Matris boyutu
2   N = 1000;
3
4   % Rastgele bir matris oluştur
5   A = rand(N, N);
6   B = zeros(N, 1);
7
8   % parfor döngüsü
9   parfor i = 1:N
10      B(i) = sum(A(i, :));
11  end
12
13  disp('Sonuç:');
14  disp(B);
15
```

Parfeval Loop

- Parfeval is used to perform asynchronous operations
- In this example we will calculate the average of a sequence of random numbers.
- We will perform the calculation in the background with parfeval, so the MATLAB session will be available before the calculation is completed.

Editor - C:\Users\onur.ozkan\OneDrive - figes.com.tr\Masaüstü\Parallel Server Example MATLAB\calculateMeanExam

calculateMeanExample.m | Perfor_For_Example.m | parsim_example.m | calculateMean.m | unt

```
% Rastgele bir sayı dizisinin ortalamasını hesaplamak için parfeval
% kullanımı Örneği

% Dizinin uzunluğunu belirliyoruz
n = 1e7;

% Asenkron olarak 'calculateMean' fonksiyonunu çaıştırıyoruz
f = parfeval(@calculateMean, 1, n);

% Görevin tamamlanmasını bekleyin ve sonucu alın
wait(f);
meanValue = fetchOutputs(f);

% Sonucu ekrana yazdırıyoruz
disp("Hesaplanan Ortalama Değer: " + meanValue);

% Fonksiyonu Tanımlıyoruz.
function result = calculateMean(n)
    data = rand(1, n);       % n uzunluğunda rastgele bir dizi oluştur
    result = mean(data);     % Dizinin ortalamasını hesapla
end
```

SPMD Blocks – Codistributed Function

- spmd (single program multiple data) enables parallel computation by sharing data between different workers
- A distributed matrix is created with the codistributed function.

Editor - C:\Users\onur.ozkan\OneDrive - figes.com.tr\Masaüstü\Optimization Example\bigdata.m

myObjectiveFunction.m | mainNoneParallel.m | mainParallel.m | bigdata.m

```
% Matris boyutu
N = 1000;

% spmd bloğu içinde paralel hesaplama
spmd
    A = codistributed.rand(N, N); % Dağıtılmış matris oluşturma
    B = sum(A, 2); % Satır toplamlarını hesaplama
end

disp('Sonuç:');
disp(B{1}); % İlk işçinin sonucunu gösterme
```

Batch usage (HPC)

- The batch command runs a specific function in the background.
- When the job is completed with fetchOutputs, the result is obtained.

Editor - C:\Users\onur.ozkan\OneDrive - figes.com.tr\Masaüstü\Optimization Example\bigdata.m

myObjectiveFunction.m | mainNoneParallel.m | mainParallel.m | bigdata.m

```
% Uzak bir iş oluştur ve çalıştır
j = batch(@() sum(rand(1e6,1)), 'Profile', 'local');

% İşin tamamlanmasını bekle
wait(j);

% Sonucu al ve göster
result = fetchOutputs(j);
disp('Sonuç:');
disp(result{1});

% İşi temizle
delete(j);
```

createJob Function

- The createJob function creates and runs jobs on the parallel pool.
- Tasks are added to the job with createTask and the job is started with submit.

Editor - C:\Users\onur.ozkan\OneDrive - figes.com.tr\Masaüstü\Optimization Example\bigdata.m

myObjectiveFunction.m | mainNoneParallel.m | mainParallel.m | bigdata.m

```
% Paralel havuz oluşturma
p = parpool('local');

% İş oluşturma
job = createJob(p);

% Görev ekleme
createTask(job, @sum, 1, {rand(1e6,1)});

% İşi çalıştırma
submit(job);

% İşin tamamlanmasını bekle
wait(job);

% Sonucu al ve göster
result = fetchOutputs(job);
disp('Sonuç:');
disp(result{1});

% İşi temizle
delete(job);
```

Usage Scenerios

Application Examples:

- Analysis of large data sets and processing.
- Tall Array Usage

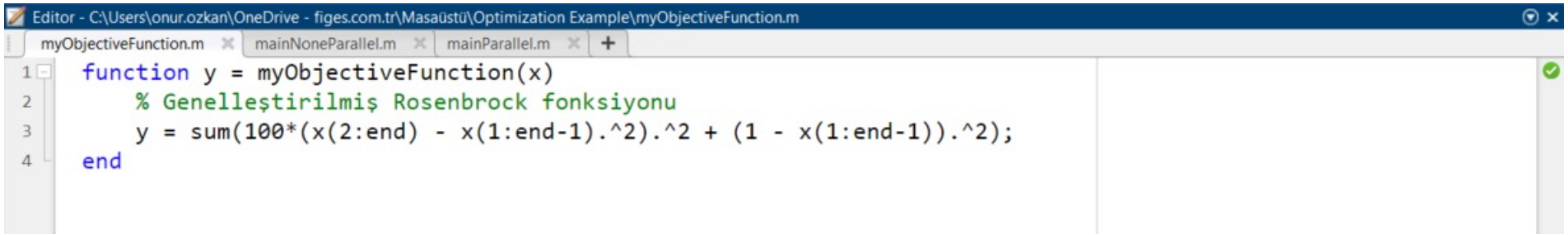
myObjectiveFunction.m | mainNoneParallel.m | mainParallel.m | bigdata.m

```
% Parallel pool oluşturma
parpool('local', 4); % 4 işçi ile paralel havuz oluşturur

% Büyük veri seti oluşturma
n = 1e7; % 10 milyon satır
d = 20; % 20 sütun
data = randn(n, d); % Rastgele sayılardan oluşan büyük veri seti

% Tall array oluşturma
tallData = tall(data);

% Zaman ölçümünü başlat
tic;

% Paralel hesaplama ile ortalama ve standart sapma hesaplama
opts = {'UseParallel', true};
meanValues_parallel = mean(tallData, opts{:});
stdValues_parallel = std(tallData, opts{:});

% Hesaplamaları yürütme
meanValues_parallel = gather(meanValues_parallel);
stdValues_parallel = gather(stdValues_parallel);

% Zaman ölçümünü durdur ve sonuçları görüntüle
time_parallel = toc;
disp(['Paralel hesaplama süresi: ', num2str(time_parallel), ' saniye']);
disp('Ortalama Değerler (Paralel):');
disp(meanValues_parallel);
disp('Standart Sapmalar (Paralel):');
disp(stdValues_parallel);
```

Usage Scenerios

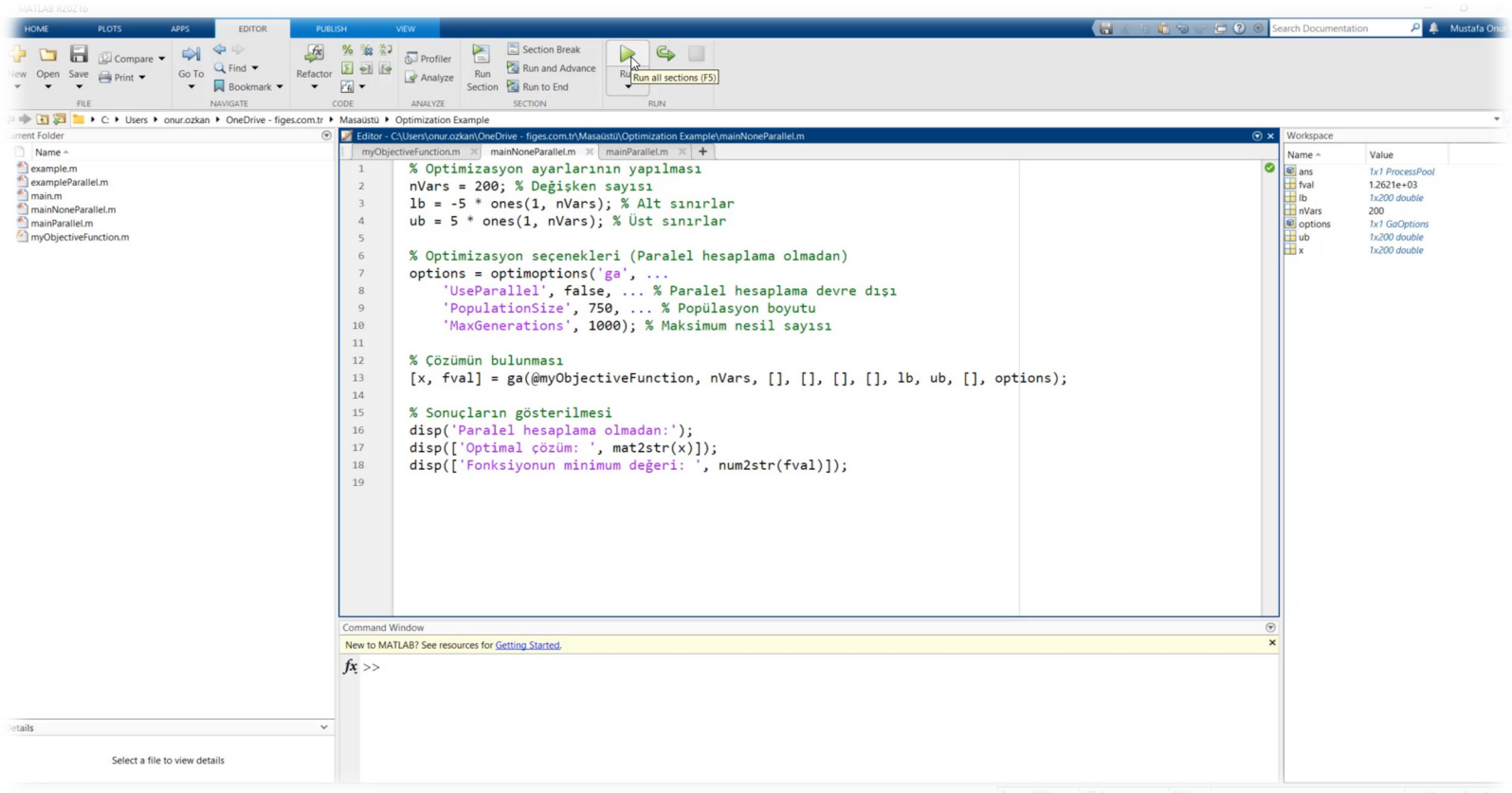
Application Examples:

- Solution of complex optimization problems.
- Rosenbrock Function



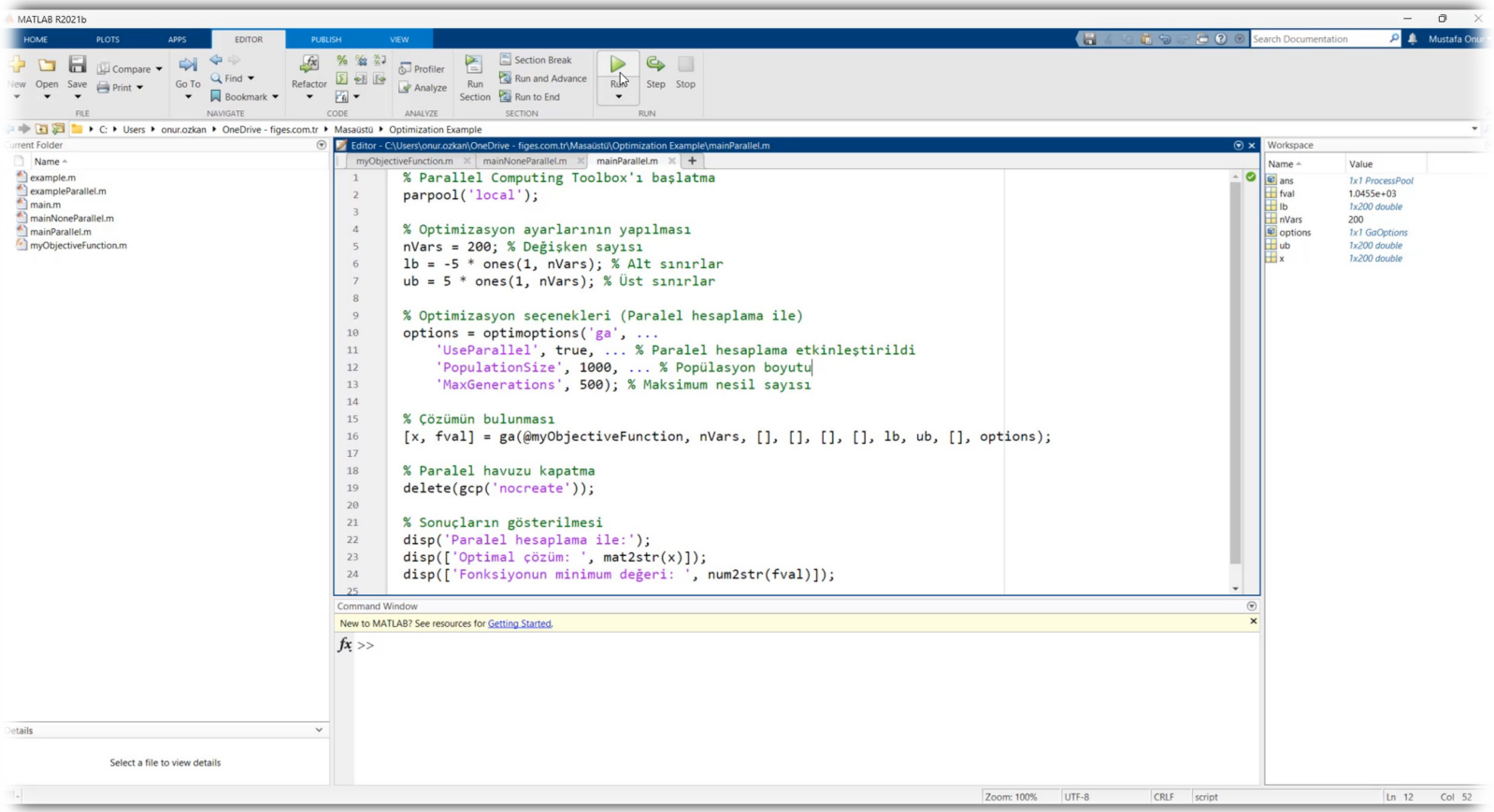
Usage Scenerios

- Without using the Parallel Computing toolbox, the relevant function produced results in 1 minute and 46 seconds.



Usage Scenerios

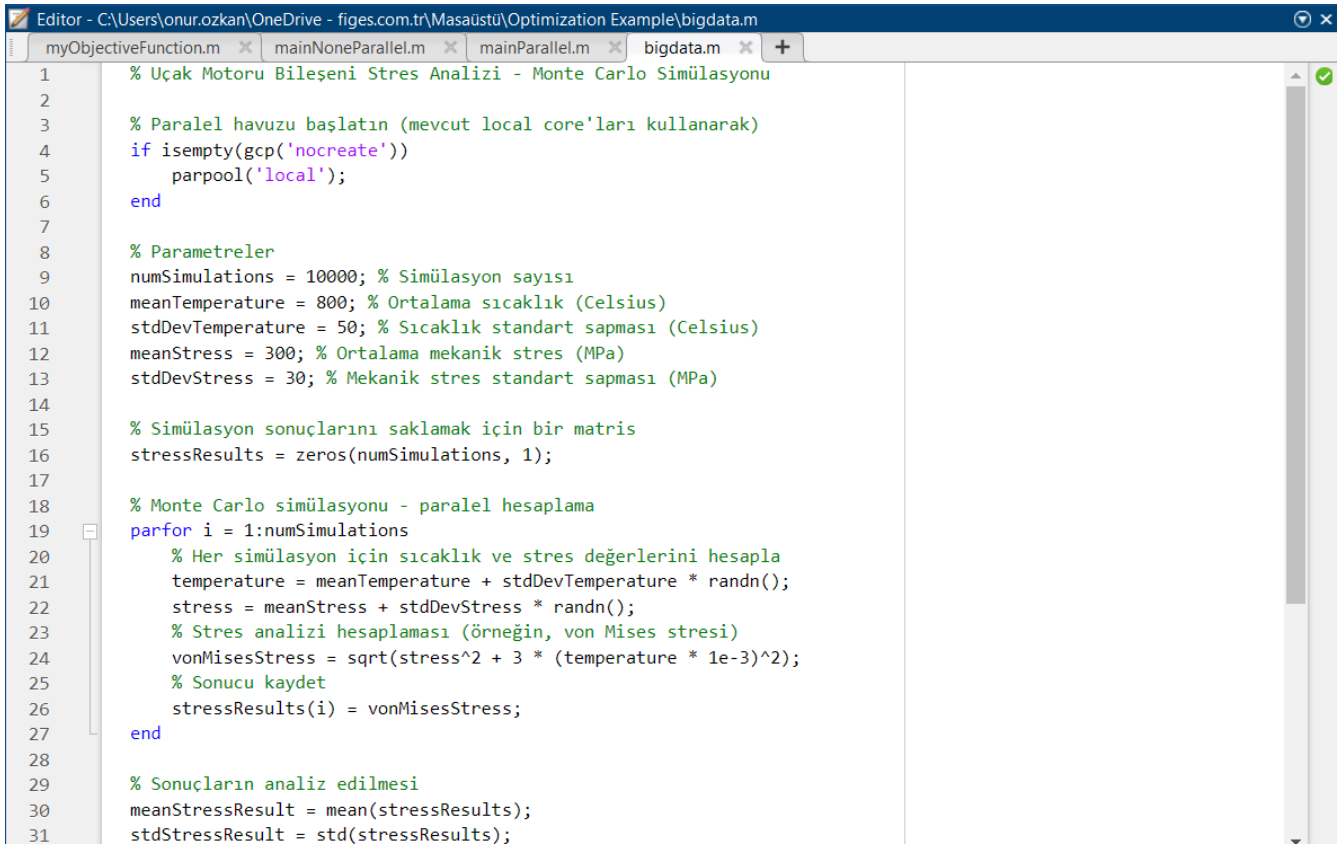
- Using the Parallel Computing toolbox, the relevant function gave results in 1 minute.



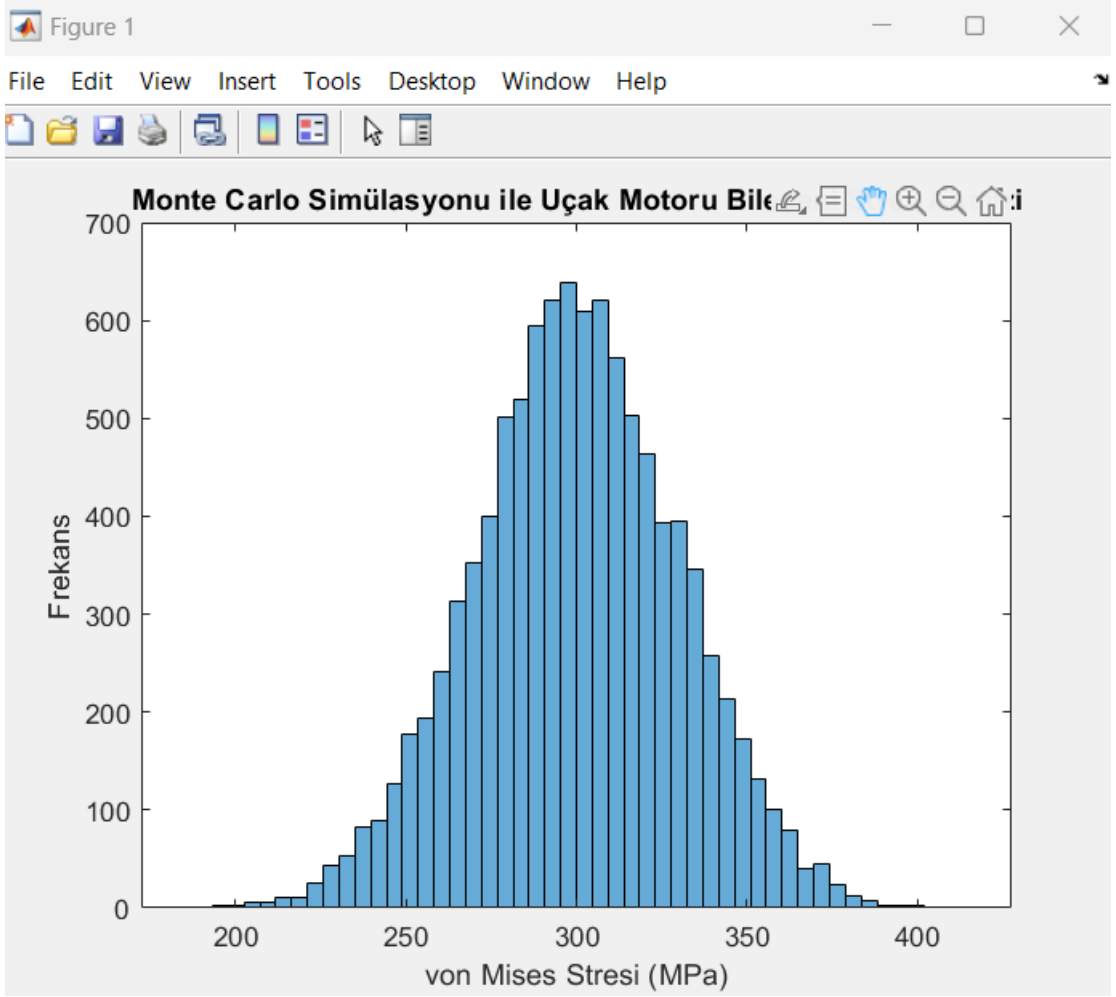
Usage Scenerios

Application Examples:

- Simulation and modeling processes.
- Monte Carlo Simulation
- Aircraft Engine Component Stress Analysis



Usage Scenerios



Usage Scenerios

Application Examples:

- Simulation and modeling processes.
- Monte Carlo Simulation
- Aircraft Engine Component Stress Analysis

```
meanStressResult = mean(stressResults);
stdStressResult = std(stressResults);

% Sonuçların görselleştirilmesi
histogram(stressResults, 50);
title('Monte Carlo Simülasyonu ile Uçak Motoru Bileşeni Stres Analizi');
xlabel('von Mises Stresi (MPa)');
ylabel('Frekans');

% Sonuçların ekrana yazdırılması
fprintf('Ortalama von Mises Stresi: %.2f MPa\n', meanStressResult);
fprintf('von Mises Stresi Standart Sapması: %.2f MPa\n', stdStressResult);

% Paralel havuzu kapatın
delete(gcp('nocreate'));
```

Usage Scenerios

Application Examples:

- Battery Management Systems (BMS)
- By simulating the temperatures and voltages in each cell in the electric vehicle battery pack, we will find the cell with the highest temperature through parallel calculation.

Editor - C:\Users\onur.ozkan\OneDrive - figes.com.tr\Masaüstü\Parallel Server Example MATLAB\BMS_Parallel_Computing.m

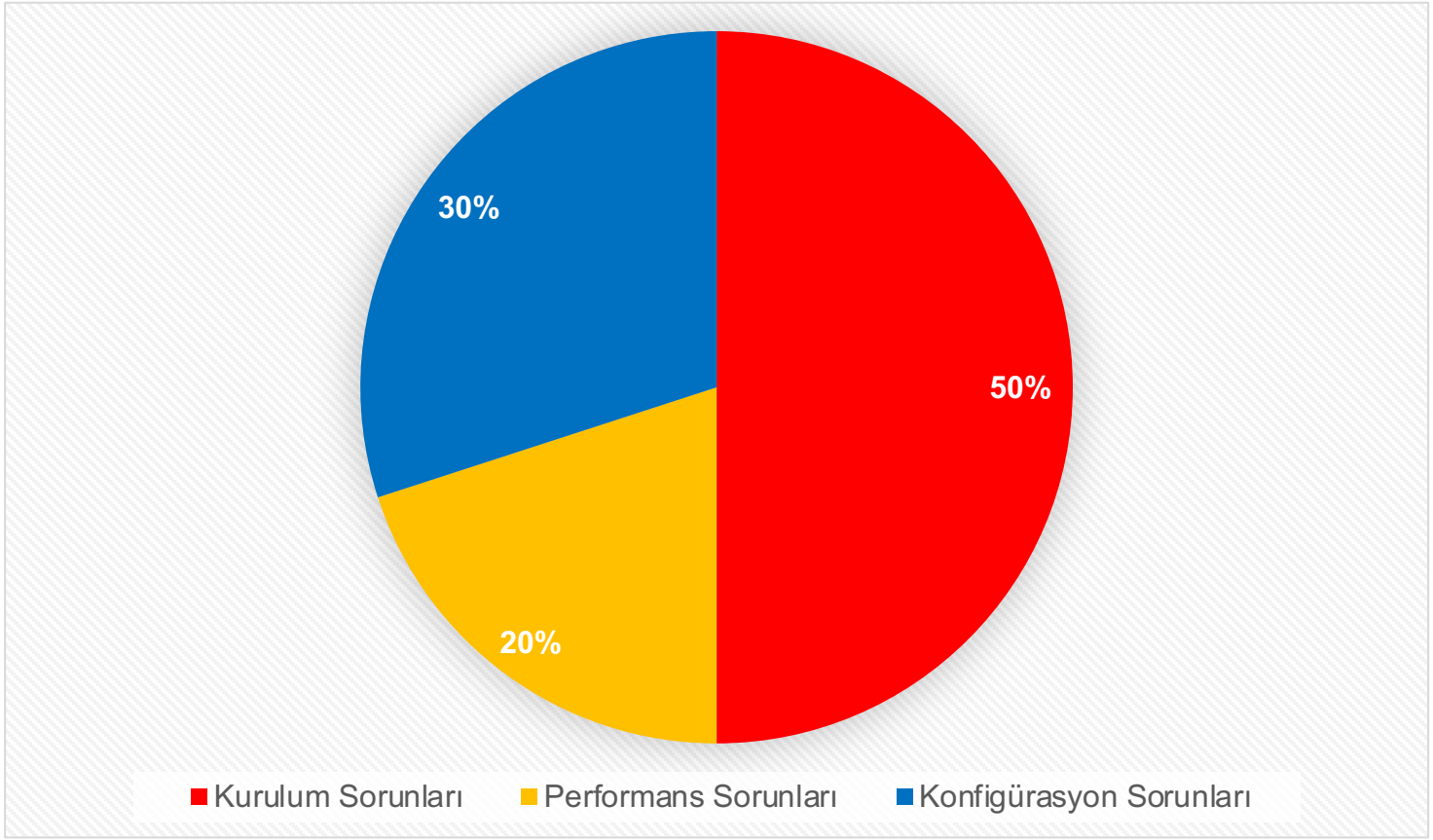
BMS_Parallel_Computing.m

```
% Script: BMS_Parallel_Computing.m
% Elektrikli araç batarya yönetim sistemi için paralel hesaplama örneği

% Hücre sayısını belirleyin
numCells = 5000;

% Hücrelerin sıcaklık ve voltaj değerlerini rastgele belirleyin
temperatureData = 30 + 20 * rand(1, numCells); % Sıcaklıklar 30-50°C arası
voltageData = 3.0 + 0.5 * rand(1, numCells);   % Voltajlar 3.0-3.5V arası

% Paralel işleme havuzunu başlatın (aktif değilse)
if isempty(gcp('nocreate'))
    parpool;
end

% En yüksek sıcaklık ve hücreyi bulmak için paralel hesaplama
maxTemperature = -inf;
maxCellIndex = -1;

% `parfor` döngüsü ile sıcaklıkları paralel olarak kontrol et
parfor i = 1:numCells
    if temperatureData(i) > maxTemperature
        maxTemperature = temperatureData(i);
        maxCellIndex = i;
    end
end

% Sonuçları görüntüleyin
fprintf('En yüksek sıcaklık: %.2f °C, Hücre ID: %d\n', maxTemperature, maxCellIndex);

% Paralel havuzu kapatın
delete(gcp);
```

Common Problems and Solutions

Graphic

- Installation Problems
- Performance Problems
- Configuration Problems

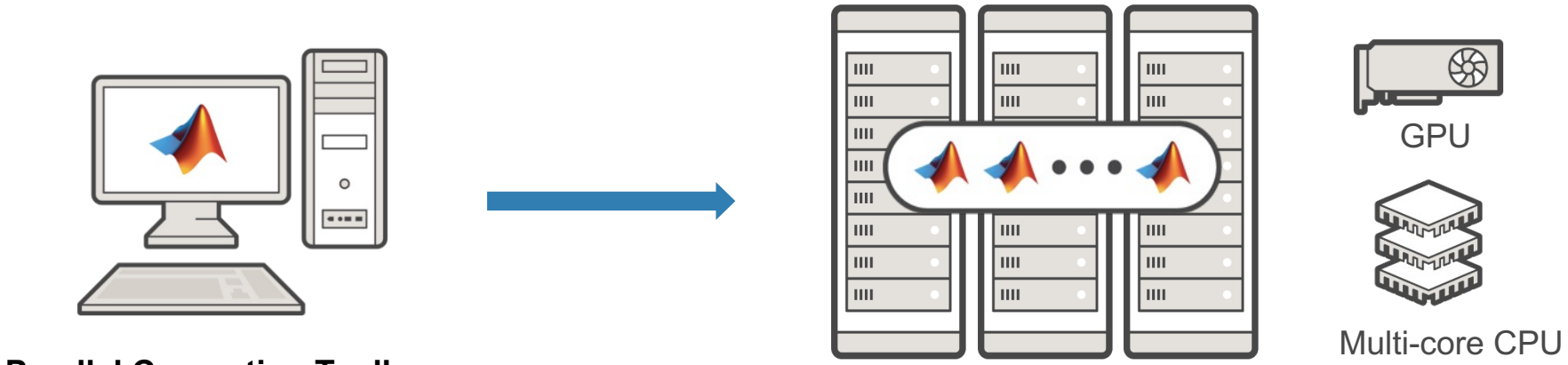


ALTAY COMPUTING SERVER SYSTEM TECHNICAL SPECIFICATIONS

System Name	ALTAI		
Processor type	AMD EPYC™ 7742	Intel XEON 8362	AMD EPYC 7543
Number of compute nodes	88	30	10
Node core count	128	64	
Node memory amount	256 GB	512 GB	1024 GB
Special nodes	-	• NVIDIA A100 80 GB PCIe • 8.8 TB scratch disk	• 4 x NVIDIA A100 80GB NVLink • 12 TB scratch disk
Performance network connection	HDR InfiniBand (200 Gbps)		HDR InfiniBand (200 Gbps)
File System	BeeGFS, 1.6 PB		
Operating System	Red Hat Enterprise Linux 8.5		

MATLAB Parallel Computing at Uhem

- When you are ready to run your MATLAB application beyond your desktop, **UHeM** has powerful HPC resources that allow you to run MATLAB to scale using MATLAB Parallel Server



- To get started:
  - https://www.uhem.itu.edu.tr/MATLAB-par/matlab.html
  - hesap@uhem.itu.edu.tr

Thank You !

Contact: onur.ozkan@figes.com.tr