Welcome to the Webinar

*Integrating Gurobi into State-of-the-Art Application Architectures*
Today’s focus:

An overview of important aspects of an application infrastructure for optimization systems.

• Solving business problems using mathematical optimization methods
• Optimization as a service
• Deployment considerations
• Common scenarios
• Licensing options
• Support
Introduction

Solving business problems using mathematical optimization methods
“Always try MIP”

• Optimization is used in many industries and across a broad range of business problems
  • Gurobi is used in dozens of industries and by over 1600 companies.
  • The reason for such broad use is the ability to rapidly solve a wide range of problem types.

• Planning usually involves a lot of “what if” analysis
  • What if costs or prices change?
  • What if customer demand changes?
  • What is the impact of adding more capacity?

• Modern MIP solvers contain a wealth of techniques, drawn from a broad range of domains
• MIP is robust with respect to changes in the underlying model
  • You can modify the problem structure (rules, conditions, logic, …)
  • You can modify data (prices, costs, weights, …)
  • You can modify goals (revenue, utilization, fairness, …)
The Optimization Workflow

- Problem Instance
- Model Generator
- Model Instance
- Gurobi Optimizer
- Solution Retrieval
- Analysis

Image: http://optano.net/en/production-planning/
Gurobi is a library

• The **Gurobi Optimizer** is a high-performance solver that can be embedded into applications that solve decision problems.

• The algorithmic details of solving optimization models are usually invisible for users
  • In many cases: „Black box“ to transform selected input data into recommendations
  • End-users are usually problem domain experts
  • No need to expose internal details (but sometimes helpful)

• **Model development**: An expert (modeller) designs the mathematical structure of the optimization model during development of the application.

• **Model generator**: The application contains a configurable software component to generate optimization model instances based on data and preferences selected by the users.
Optimization as a service
Model solving is a service

• Modern application architectures divide components into decoupled functional units (services).

• An optimization service provides the required
  • functional capabilities (model solving) on
  • its input data (the model instance) using a
  • well-defined interface (API).

• Applications that solve optimization models as part of their functionality are optimization service clients.

• A service can be local or remote (e.g., a web service). Remote services are usually available to multiple clients at the same time.

• It is up to a software architect to decide which functionality is encapsuled into services.
Different levels of abstraction

Optimization services can be implemented with different levels of abstraction:

• **General-Purpose Service**

  The service accepts a model instance and returns the results when finished. There can also be functions to monitor progress and control job termination.

  **Pro:** Most flexible way to solve any kind of model  
  **Con:** Requires model building capabilities on the client side

• **Domain-Specific Service**

  The service accepts a set of data for a well-defined problem type and handles everything in the background (model generation, solving, solution retrieval). When finished, the service returns results – may also include post-processing.

  **Pro:** Hides most complexity in the background  
  **Con:** Limits service clients to certain problem types
Gurobi Compute Server

• Gurobi Compute Server is a high-performance optimization service implementation based on state-of-the-art web standards.

• Key factors when comparing Compute Server to your own service implementation:
  
  • **No interface change**: The client application uses the exact same API to solve an optimization model compared to solving on a local machine. No need to rewrite an existing application.

  • **Flexible deployment**: Client/server separation is optional. The decision on where a model should be solved (locally or remote) can be changed at any time.

  • **Scalability**: Gurobi Compute Server can be deployed on a single server machine or within a cluster of multiple powerful machines that can be accessed remotely. Features for job management and high-availability (e.g., load balancing, queuing and failover) are already implemented.

  • **Performance**: Compute Server is a completely integrated into the Gurobi Optimizer. No need for any intermediate software layer that consumes additional memory or CPU resources.
Deployment Considerations
Usage patterns

• Depending on the application the usage patterns can be quite different:

  • **Single-Threaded vs. Multi-Threaded**

    Gurobi was designed to efficiently (and deterministically) solve models on state-of-the-art processors with multiple cores. Most optimization models benefit from parallelism. However, for certain model types (and especially very small models) parallelism does not necessarily accelerate performance.

  • **Sequential vs. Parallel**

    Some optimization problems can be split into smaller subproblems. Sometimes subproblems can be solved independent of each other in parallel. For some problems, the problems are connected so multiple models can only be solved in parallel.

  • **Single-Machine vs. Distributed Algorithms**

    For certain models types (especially MIP models with a large search tree) performance can be greatly improved if multiple machines work on the same model in parallel. Gurobi contains a set of Distributed Algorithms for these models.
People often contact Gurobi support with hardware questions:

• „I'm going to invest into new hardware. What is the best machine for my optimization application?“
• „Will my model solve faster with more RAM?“
• „Which CPU models do you recommend?“
• „How many models can I solve in parallel on this machine?“

Answer: „It depends“

If you are solving a large MIP model in parallel, you'll get the best performance from a system with
• the fastest possible clock rate and
• 4 channels per socket of the fastest available memory.

There is no hardware recommendation for all models. You are welcome to support@gurobi.com to discuss your specific models.
Network Communication

• The network communication overhead is usually negligible when connecting servers on the same local network.

• However, be aware of potential network issues when retrieving data or accessing Gurobi Compute Server machines
  • High latency
  • Low bandwidth
  • Stability

• Latency can become an issue when a lot of messages are sent over the network.
Programming Languages

• You don’t have to choose a specific programming or modeling language to use Gurobi.

• Different Gurobi customers use different interfaces
  • Python
  • Java
  • C++
  • .NET (C#, Visual Basic, …)
  • MATLAB,
  • R
  • ...

• Gurobi is committed to all of these interfaces
• Our APIs are just a thin layer on top of the high-performance library.
Operating Systems

- Gurobi supports all major operating systems:
  - Windows
  - Linux
  - Mac

- There is no recommendation: Choose whatever platform seems right for you.

- Operating systems can be mixed between client and server, e.g. a Windows client can connect to a Linux machine running Compute Server.
Different Levels of Virtualization

Modern CPUs and operating systems support different levels of virtualization:

• **Hardware Virtualization** (Virtual Machines)
  Gurobi is used on virtual machines by many customers. On modern CPU architectures, the overhead of running multiple independent virtual machines on a single physical hardware is small.

• **Desktop Virtualization** (Terminal Server)
  Virtualized desktops allow to run multiple applications on the same machine by multiple users at the same time (remote sessions).

• **OS-Level Virtualization** (Containers)
  Containers are an abstraction at the application layer that packages code and dependencies together. Multiple containers can run on the same machine and share the same operating system with other containers, each running as isolated processes in user space.

Gurobi can be used at every virtualization level. For performance reasons it is important to consider that optimization runs need to share hardware resources with other processes on the same physical machine.
Benchmarking & Tuning

• Benchmarking is very important during the development of an optimization application.

• Important goals:
  - Estimate performance (Runtimes to build and solve your models)
  - Identify bottlenecks (especially data preparation and transformation)
  - Find the right hardware that supports your usage patterns

For MIP models, it is nearly impossible to predict solving runtimes based on the model size.
Common Scenarios
Single-Machine Deployments

• Many Gurobi customers solve optimization models without client/server separation on the application side.

• Examples:
  • Small/fast models, „easy“ model types
  • Low-priority optimization jobs
  • Strict hardware resource allocation (example: dedicated CPU cores/memory)

• Still viable for many use cases
  • Very regular usage patterns
  • No upscaling required
Frontend / Backend

• Separation of hardware between application and Gurobi
  • Frontend: The main application runs on one or more other machines (application server)
  • Backend: Optimization models are solved on a dedicated machine (optimization server)

• Benefits:
  • No concurrency between resources on the same physical hardware
  • Improved scalability
  • Easy to implement using Compute Server

• Compute Server is not necessarily required. The optimization server can run its own backend service implementation (with a single-machine license).
Containerized Applications

- Containerized applications share a common operating system kernel
  - No need for each application to run within its own operating system.
  - An application can be deployed in a matter of seconds
  - Fewer resource consumption compared to hardware virtualization.

- Most popular systems:
  - Docker (container management)
  - Kubernetes (orchestration platform)

- Good news: From Gurobi’s perspective a Docker container is just a 64-bit Linux system.
  - It works out of the box
  - Can be used with any client license (no single-machine licenses)
Cloud Applications

• More and more customers are moving application infrastructure into a cloud system.

• Public cloud providers have made it very easy to build whole IT infrastructures in the cloud:
  • Virtual networks
  • Virtual network components (router, load balancer, etc.)
  • Virtual computers

• Powerful hardware (exclusive and guaranteed) is available within seconds.
  • Certain machine types work very well for optimization
  • Requires control over machine termination (e.g., no „spot instances“)

• All Gurobi licenses can be used in a cloud environment
Licensing

Single-Machine, Compute Server, Floating, Instant Cloud
Single-Machine licenses

• A **Single-Machine** license allows to run Gurobi on a single virtual or physical computer.

  • **Unlimited-User** license
    • Unlimited uses (concurrent runs)
    • Pricing is based on the number of physical CPU cores
    • Not available for Docker

  • **Named-User** license:
    • Unlimited uses (concurrent runs)
    • For a single identifiable person
    • No limit on the number of physical CPU cores
    • Not available for Docker

  • **Compute Server** license: Use Gurobi Compute Server with an unlimited number of clients
    • A limit on the maximum number of parallel jobs can be configured
    • Client machines don't need a separate license
    • Pricing is based on the number of physical CPU cores
Floating licenses

A floating license allows to run a specified number of simultaneous Gurobi runs in parallel on any machine in the network.

• When using a floating license, a program that calls the Gurobi Optimizer must obtain a token from a Gurobi token server before it can solve an optimization model.
• The token server is a process that runs in the background, handing out available tokens to programs as they request them.
• Finally, each client for the token server will need to create a token server client license to allow client programs to find the token server.
Gurobi Instant Cloud

- You don’t have to buy and manage your own hardware to run Gurobi. The **Gurobi Instant Cloud** is a simple and cost-effective way to get up and running with powerful Gurobi optimization software running on cloud systems.

- **Key features:**
  
  - **Easy to use:** Simply install Gurobi software on a local machine, download your license from the Cloud Manager (cloud.gurobi.com), and the Gurobi Instant Cloud connects any machine to the cloud.
  
  - **Security:** End-to-end encrypted connection between client and server.
  
  - **Scalability:** Define your own machine pools based on your performance needs - from small, single-machine setups to powerful Compute Server clusters.
  
  - **Local:** Choose from different regions around the globe to minimize network communication overhead between client and server machines.
  
  - **Full control:** Start, stop and manage machines from anywhere using a REST API.
  
  - **Pay for what you use:** Cloud pricing is simply based on machine uptime and hardware power.
Licensing for Docker

• Single-Machine licenses cannot be used with Docker but all other client licenses:

  • **Compute Server** license:
    Models are solved on one or more Compute Servers machines outside of the Docker container.

  • **Instant Cloud** license
    Models are solved on one or more Compute Server machines running in the cloud outside of the Docker container.

  • **Floating** license
    Models are solved locally within the Docker container.
Support
How we support you

• Chosing the right infrastructure for your application may also require a lot of planning and discussion.

• No need to do this on your own: The Gurobi Support team is here to assist you during a product evaluation: support@gurobi.com

• We provide direct access to optimization experts with PhDs and years of experience with commercial models:
  • Technical assistance
  • Benchmarking & Tuning
  • Architecture discussions
  • Hardware recommendations
Further Reading

• We have touched a lot of topics on the surface in this webinar.
• Check out our recent webinars for more detailed information: http://www.gurobi.com/resources/seminars-and-videos/seminars-videos

• Still not sure if and how optimization can help your company?
  • How Optimization Modeling Creates Value for an Organization
  • Building the Business Case for Optimization

• Learn about the latest advances in technology and performance
  • What's New in Gurobi 8.0
  • Compute Server and Cloud

• Learn about modeling & tuning
  • Modeling with Python
  • Introduction to Performance Tuning