VORTECH

WHITEPAPER

Eight ways to speed up your computing software.

Compute more in less time.

Content

Introduction 3

1. Performance-aware programming 4

2. Use parallelism in standard hardware 5

3. Use GPGPUs 6

4. Use a different solver 7

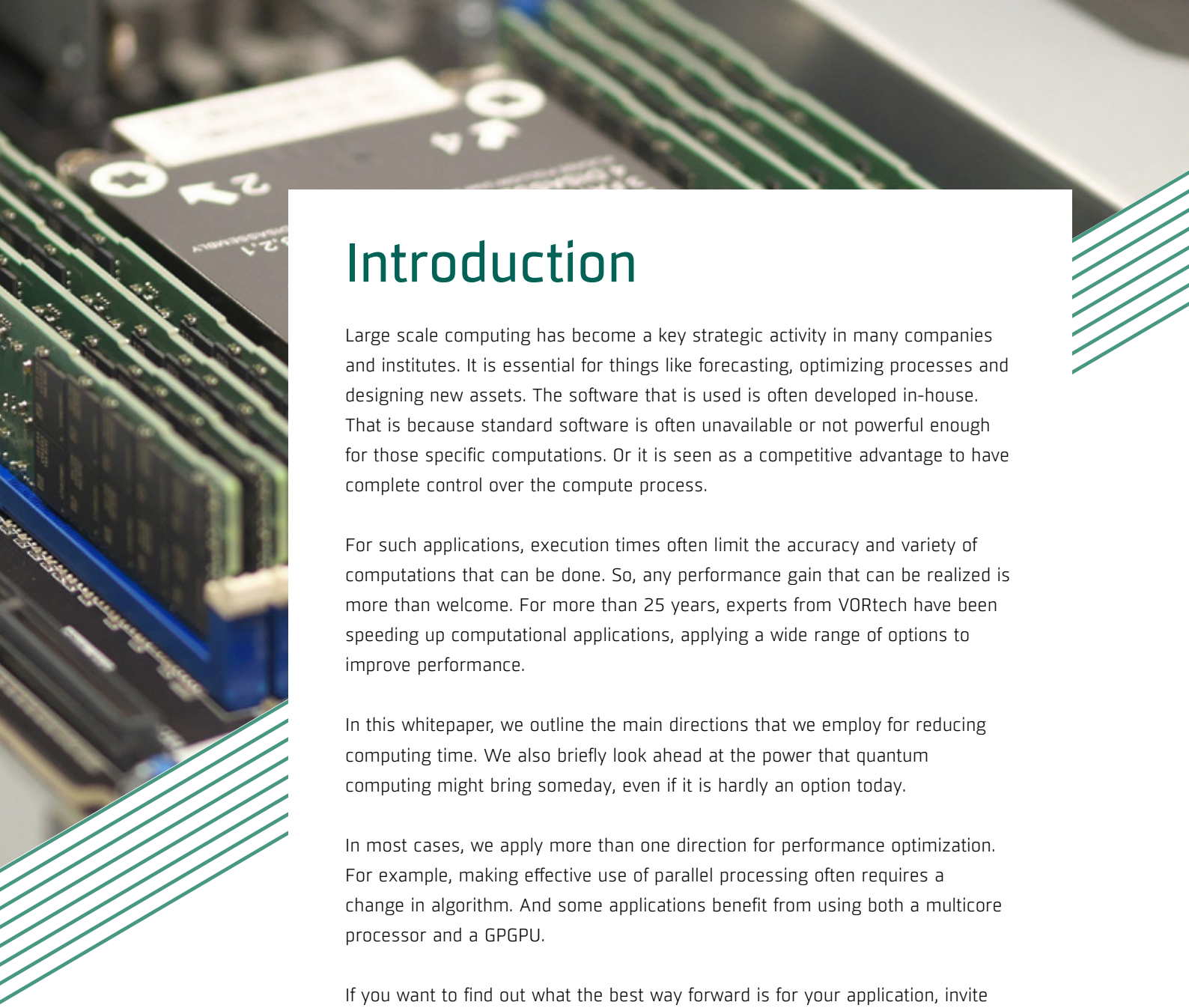
5. Reducing model complexity 8

6. Machine Learning 9

7. High-Performance Computing 10

8. Quantum computing 11

About VORtech 12



Introduction

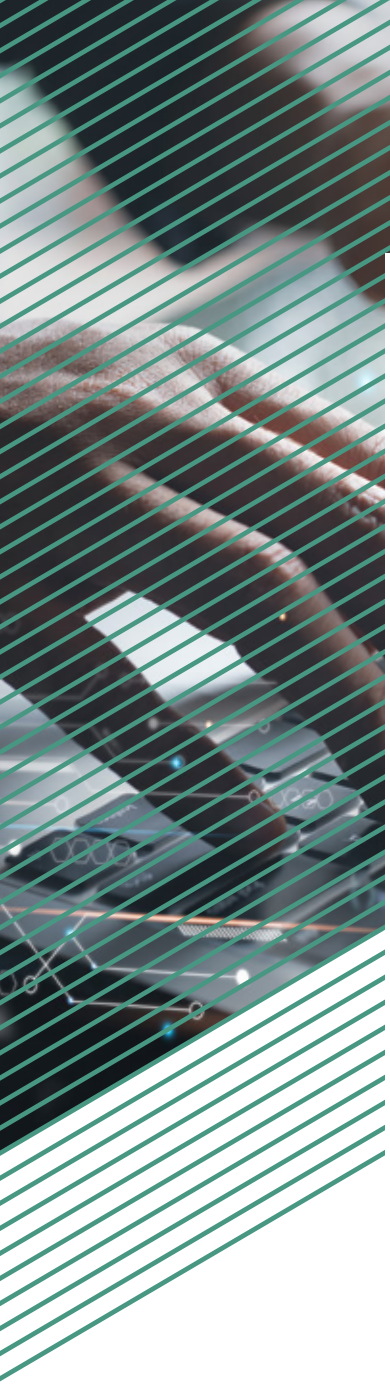
Large scale computing has become a key strategic activity in many companies and institutes. It is essential for things like forecasting, optimizing processes and designing new assets. The software that is used is often developed in-house. That is because standard software is often unavailable or not powerful enough for those specific computations. Or it is seen as a competitive advantage to have complete control over the compute process.

For such applications, execution times often limit the accuracy and variety of computations that can be done. So, any performance gain that can be realized is more than welcome. For more than 25 years, experts from VORtech have been speeding up computational applications, applying a wide range of options to improve performance.

In this whitepaper, we outline the main directions that we employ for reducing computing time. We also briefly look ahead at the power that quantum computing might bring someday, even if it is hardly an option today.

In most cases, we apply more than one direction for performance optimization. For example, making effective use of parallel processing often requires a change in algorithm. And some applications benefit from using both a multicore processor and a GPGPU.

If you want to find out what the best way forward is for your application, invite us to do a model scan. This will give you actionable advice that can guide your developers in optimizing the application.



1. Performance-aware programming

How does it work?

Naïve programming can cost you a lot of performance. A bit of programming discipline can already provide quite some performance gains. Reducing function calls, avoiding unnecessary memory allocations, loop reordering and intelligently choosing between reusing or recomputing results are standard ways to reduce computing time.

Potential speedup

The speedup obviously depends on how good the original programming already was, but a speedup by a factor of 2 or more is not unusual if the original program was not yet optimized.

Advantages

- This can be done by any developer with a sense of how the program is executed.
- Usually, the code changes are small and often the code quality becomes even better.
- Programming guidelines can help to establish performance-aware programming habits.
- Performance analyzers are widely available to help find the performance issues.

Disadvantages

- It requires developers that are aware of the performance consequences of their choices and understand both the compiler and the hardware.
- When pushing for all the performance that the hardware can provide, it can be challenging to have both performance and readable code.

2. Use parallelism in standard hardware

How does it work?

The processing unit in most computers typically consists of multiple processors that each have multiple computing cores. This makes it possible to execute parts of the application in parallel. The number of cores per processor is typically 4 or 8 but can go up to 128 in top-of-the-line models. The number of processors in a system is usually 2.

Potential speedup

Maximum speedup depends strongly on the total number of cores of the hardware and on the application. Typically, the application performance starts increasing linearly when using more cores, but after a certain point the parallelization becomes less efficient, and any additional cores will have less of an impact. For example, an application might see a 3x speedup from using 4 cores, but only a 7x speedup from using 16 cores.

Advantages

- No exotic hardware is needed; it just uses the available standard hardware better.
- Some performance gains can be obtained with relatively simple code changes that are easy to understand.
- Significant refactoring of the software is not always needed.
- Cloud providers typically offer the latest generation multicore processors, giving low-entry access to high performance compute.

Disadvantages

- Performance gains are limited to the CPU resources of a single server.
- Significant refactoring of code may be necessary to get the most out of the hardware. The aim is to maximize the parallelism in the application and minimize the bottlenecks that can occur when multiple processes are all accessing memory over the same bus.

3. Use GPGPUs

How does it work?

General Purpose Graphical Processing Units (GPGPUs) have many small computing cores that act in lockstep to perform an operation on many different data-items simultaneously. If your application is characterized by doing the same computation for a huge number of data points, then a GPGPU may give significant speedup. The number of computing cores in high-end GPGPUs can go up to several thousand.

Potential speedup

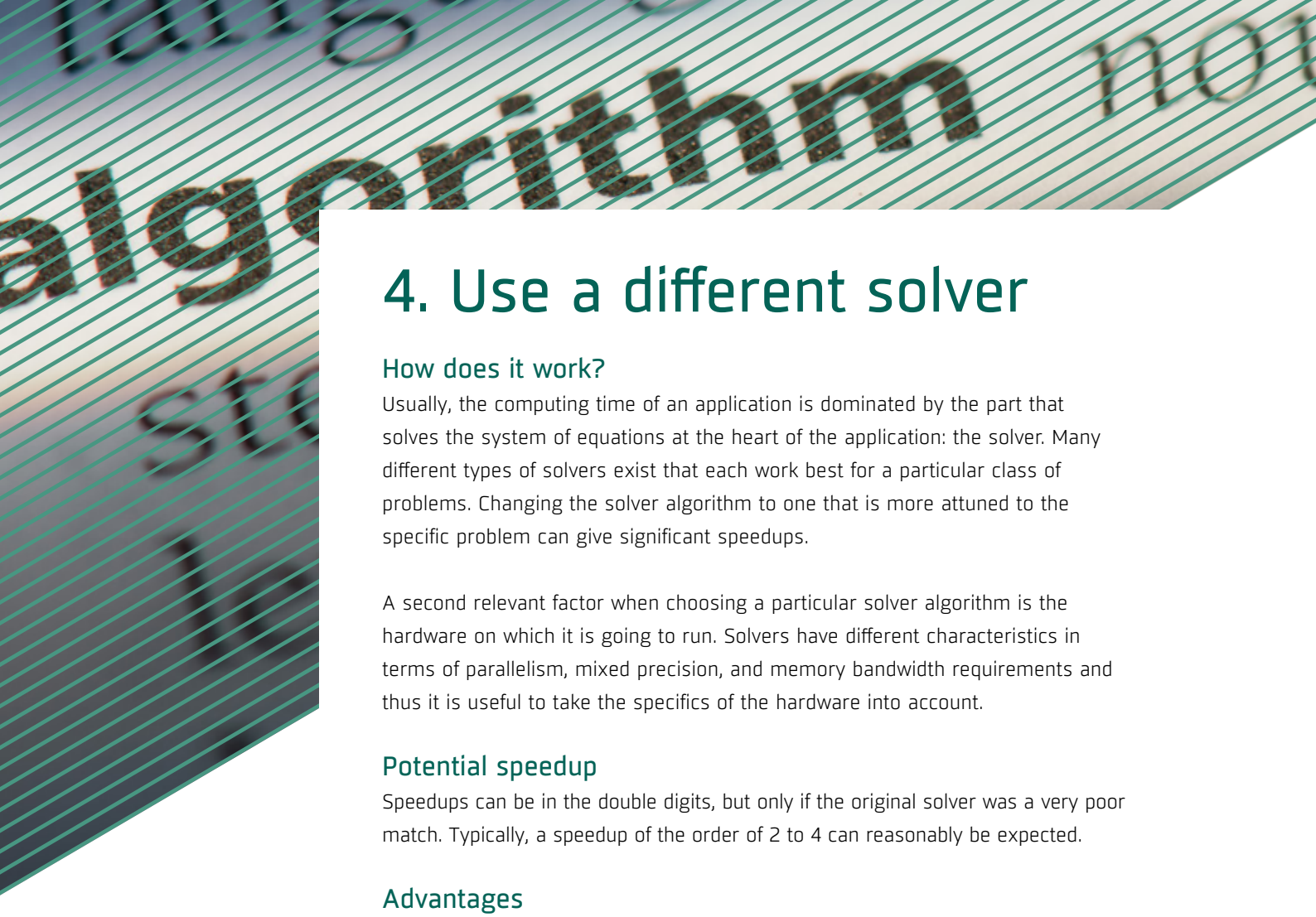
The speedup over a CPU computation can easily be around 10 but depends strongly on the degree of parallelism in the application. Typical speedup is in the order of 4.

Advantages

- Potentially a high speed up, even if this is only attainable for very specific applications.
- Energy-efficient: GPGPUs use less energy per computation than ordinary processors.
- Even higher performance and energy efficiency is possible when using lower precision arithmetic.
- GPGPUs are widely available and easy to integrate in your system.
- GPGPUs are available on public clouds.

Disadvantages

- Speedup is limited for most applications.
- Making use of GPGUs usually requires a significant change in the code and introduces constructs that are less familiar to many developers.
- For some applications, significant gains are only achieved when the whole application is ported to use the GPU. This is because transferring data between CPU and GPU is relatively slow.



4. Use a different solver

How does it work?

Usually, the computing time of an application is dominated by the part that solves the system of equations at the heart of the application: the solver. Many different types of solvers exist that each work best for a particular class of problems. Changing the solver algorithm to one that is more attuned to the specific problem can give significant speedups.

A second relevant factor when choosing a particular solver algorithm is the hardware on which it is going to run. Solvers have different characteristics in terms of parallelism, mixed precision, and memory bandwidth requirements and thus it is useful to take the specifics of the hardware into account.

Potential speedup

Speedups can be in the double digits, but only if the original solver was a very poor match. Typically, a speedup of the order of 2 to 4 can reasonably be expected.

Advantages

- The total number of computations is reduced, leading to a more energy-efficient application.
- Depending on the structure of the application it is sometimes enough to just call a different solver routine. So, the code changes can be small.
- In some cases, a more appropriate solver can also be more robust and/or accurate.

Disadvantages

- Introducing a better solver requires specific expertise and sometimes also research effort.
- Depending on the quality and architecture of the code, significant refactoring may be necessary.

5. Reducing model complexity

How does it work?

Many computing codes are based on a mathematical model that is discretized using, for example, finite elements or finite differences. To reach a high accuracy, many discretization points are needed. However, such high accuracy is not always needed. In that case, an approximate model with less degrees of freedom can be constructed. This is called a reduced order model. Rigorous mathematical methods ensure that the reduced order model is close to the full model.

The reduced order model can be used for instance in an optimization process to get close to the optimal point. The full model then takes over for the final stretch to the exact optimum.

Potential speedup

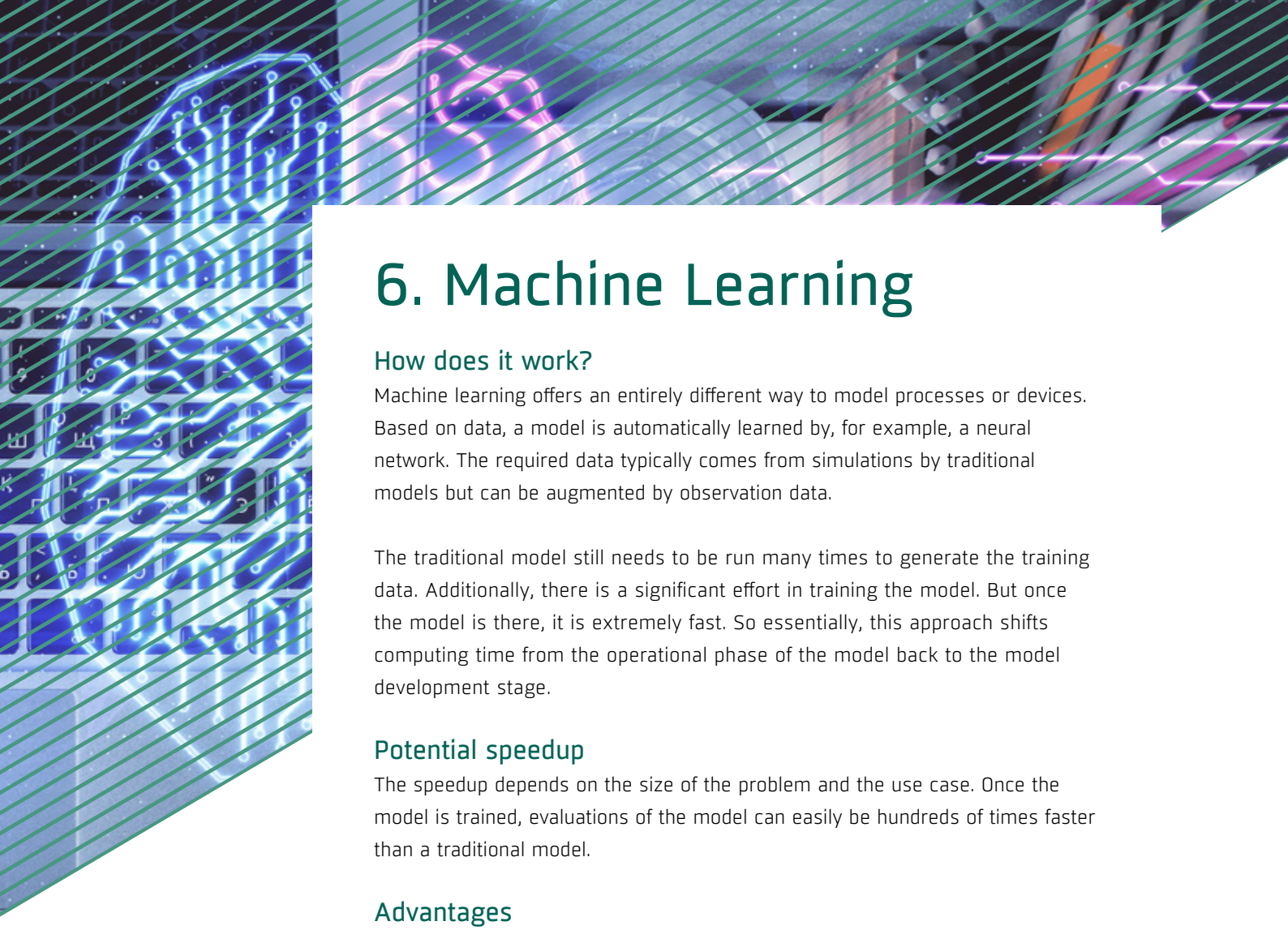
The speedup depends on the size of the problem. Typical speedup is somewhere between 4 and 20.

Advantages

- The number of computations is reduced so that not only the speed is increased but also energy consumption is reduced.
- The inaccuracy of the reduced order model can be controlled, so a conscious choice can be made.

Disadvantages

- Constructing a reduced order model is often complex. Relatively simple methods exist but these require many initial computations to gather the information needed for the reduced model.
- The model is less accurate, which is not always acceptable.
- The software complexity increases because the original and reduced complexity model must both be maintained.



6. Machine Learning

How does it work?

Machine learning offers an entirely different way to model processes or devices. Based on data, a model is automatically learned by, for example, a neural network. The required data typically comes from simulations by traditional models but can be augmented by observation data.

The traditional model still needs to be run many times to generate the training data. Additionally, there is a significant effort in training the model. But once the model is there, it is extremely fast. So essentially, this approach shifts computing time from the operational phase of the model back to the model development stage.

Potential speedup

The speedup depends on the size of the problem and the use case. Once the model is trained, evaluations of the model can easily be hundreds of times faster than a traditional model.

Advantages

- The speedups during the use of the model can be spectacular.

Disadvantages

- Extensive knowledge of machine learning is needed to build a proper model.
- The model may be less reliable as its correctness is often hard to prove.
- Care must be taken in the choice of training data, as it heavily influences the quality of the trained model.
- Building the model from simulation results still requires many computations to generate the training data and subsequently train the model.

7. High-Performance Computing

How does it work?

The term high-performance computing (HPC) refers to the use of systems with large amounts of processors. An HPC-system typically has dozens up to thousands of processors, often including GPU-nodes, connected through a high-speed network to allow efficient data-exchange.

Using these systems well requires that the application consists of a set of cooperating processes that exchange data with each other, usually through explicit messages. If an application is not designed for HPC from the start, adapting it is usually a significant effort. But when model reduction or machine learning techniques don't work, using the big hardware is the best way to go for the really big computations. That may involve switching to another numerical algorithm to reduce the overhead due to communication between the compute nodes. Even if the alternative algorithm is less efficient on a single computer, it can be much more efficient when running it in parallel on many compute nodes.

Potential speedup

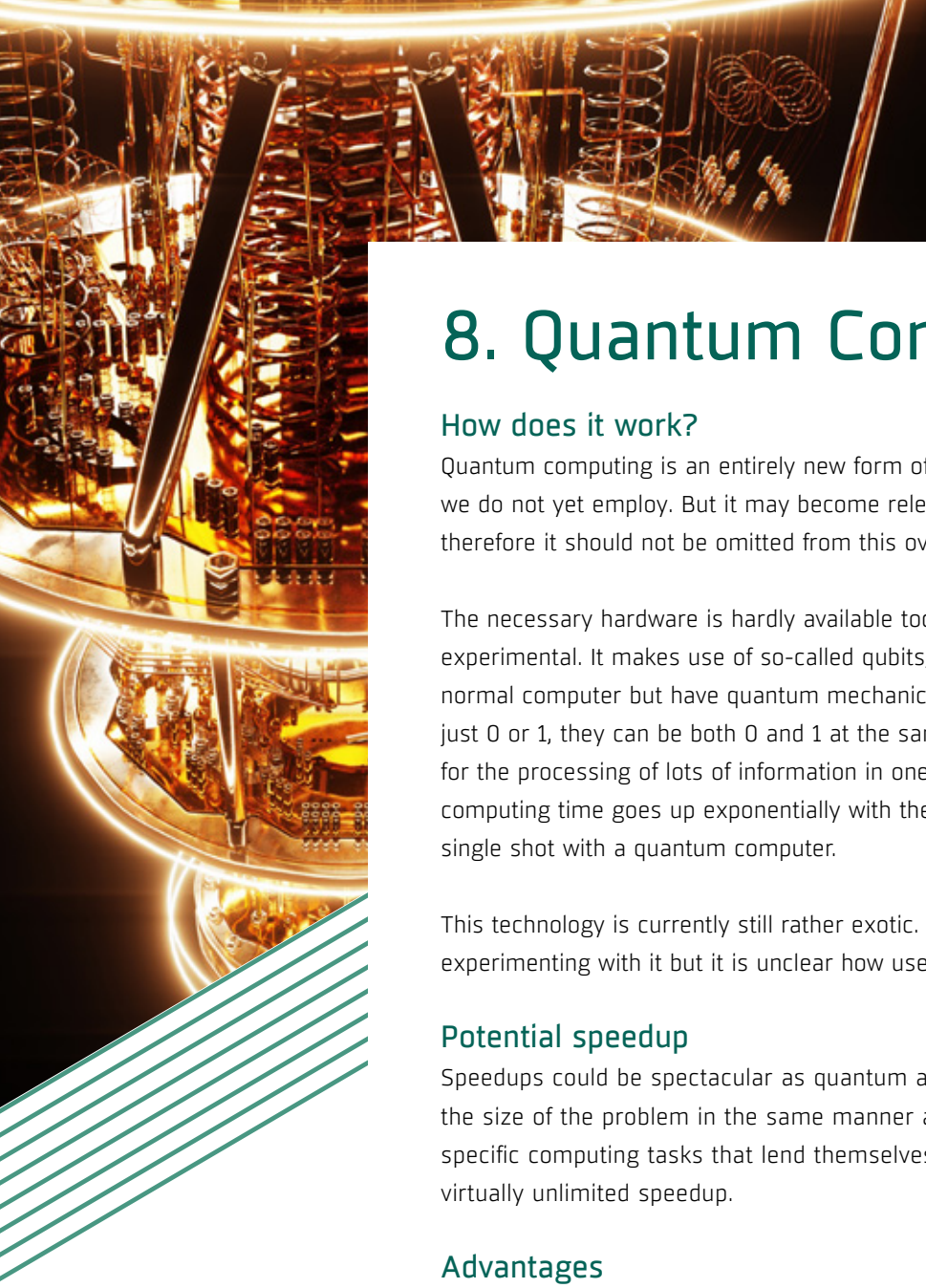
The maximum speedup is obviously bound by the number of processors in the system and by the parallelism in the application A well-designed application can scale up to hundreds of processors and reach a speedup that is a significant fraction of the number of processors that is employed.

Advantages

- Large speedups can be attained.
- Large volumes of data can be handled.
- A well-designed application can scale from using a few processors for small jobs to many processors for large jobs.
- High-performance computing systems can be accessed through HPC-providers, so usually no proprietary hardware is needed.

Disadvantages

- Applications need to be specially designed or adapted to make good use of the hardware.
- HPC-systems are usually shared with other users so larger compute jobs may have to wait for an available timeslot.
- The use of high-performance computing systems can be relatively expensive.
- When using a remote system, uploading and downloading large volumes of data can be cumbersome.



8. Quantum Computing

How does it work?

Quantum computing is an entirely new form of computing which, to be honest, we do not yet employ. But it may become relevant in the coming years and therefore it should not be omitted from this overview.

The necessary hardware is hardly available today and is mostly still very experimental. It makes use of so-called qubits, which are remotely like bits in a normal computer but have quantum mechanical properties. Rather than being just 0 or 1, they can be both 0 and 1 at the same time. This property allows for the processing of lots of information in one go. Algorithms for which the computing time goes up exponentially with the problem size can be done in a single shot with a quantum computer.

This technology is currently still rather exotic. Large companies are said to be experimenting with it but it is unclear how useful the results really are.

Potential speedup

Speedups could be spectacular as quantum algorithms do not scale with the size of the problem in the same manner as traditional algorithms do. So specific computing tasks that lend themselves to quantum computing can have virtually unlimited speedup.

Advantages

- Quantum computing allows computations of a complexity that cannot possibly be done on traditional computers.

Disadvantages

- Availability of hardware is still extremely limited, and the hardware is unreliable.
- Programming quantum computers requires entirely different skills than traditional computers.
- These skills are scarcely available.

VORTECH

About VORtech

VORtech is a company of scientific software engineers. Our experts combine a passion for software development with in-depth knowledge of engineering and mathematics. They always work in close collaboration with employees of the client, who are experts in their specific domain. Together with these experts, we bring everything to the table that is needed to create high-tech computational applications.

VORtech offers the full spectrum of expertise needed to create high-quality computational software. The basis is craftmanship in software development in a wide range of programming languages, from Fortran and C to C#, Python and Matlab. On top of that, we have experts on high performance computing, data-model fusion, machine learning and web development.

A typical engagement with VORtech starts with a model scan, where we analyze the client's software and advise on the issues that the client has with it. Next, we usually do a short project to demonstrate our skills, solving one concrete issue. After that, VORtech often becomes a strategic partner for providing our specific skills, where our involvement scales up and down with the needs of the client.

Our clients are primarily large companies, research institutes and government agencies. Our role is typically to bring software from a low TRL-level, often coming from the research department, to the high TRL-levels that are needed for operational use.