User Transparent Parallel Image Processing

Seinstra, F.J.

Publication date
2003

Chapter 1

Introduction

"The fact that the pieces do fit together [...] is something you might miss from focussing too closely on one aspect of science."

John Gribbin - *Almost Everyone's Guide to Science* (1998)

Throughout history, mankind has had an ever growing desire for increased efficiency. Irrespective of the origin of the desire, its manifestations are manifold. For example, the desire to efficiently disseminate ideas and information from a single source to a large and far-ranging audience, directly led to Gutenberg's invention of the printing press*. In the Mid-Eighteenth Century, the desire for large-scale production resulted in the application of power-driven machinery to manufacturing — and the Industrial Revolution. More recently, the need for automated processing of scientific problems, and the handling of large amounts of data, led to the advent of the Information Age.

Once its raison d'être is demonstrated, high-speed machinery is constantly being improved upon for ever increased efficiency. A good example is the development of successive generations of trains. When the English inventor Richard Trevithick introduced the steam locomotive on 21 February 1804 in Wales, it achieved a speed of 8 km/h. In 1825, Englishman George Stephenson introduced the world's first workable passenger train, which steamed along at 24 km/h. Today, the fastest passenger trains fly down the tracks at a speed of approximately 550 km/h.

Part of the success of such technologies stems from the fact that successive performance improvements generally did not result in increased user requirements. At any time, a passenger could get onto a train, sit down pleasantly, get some sleep, read a newspaper, do some work, and get off at any station, *without ever having to worry about the actual running of the train*. If, in contrast, it would have been required

*For reasons of completeness, the author would like to stress that in his city of birth (Haarlem, The Netherlands) many still consider Laurens Jansz. Coster to be the true inventor of printing [160].

from anyone travelling the latest zenith in high-speed train design to have expert knowledge regarding the train's locomotion, passenger numbers would have dropped dramatically. Unfortunately, it is exactly this problem that can be observed with respect to the latest developments in high performance computer systems.

As stated, the Information Age has seen major breakthroughs in the automated processing of scientific problems. Today, ever more complex problems are being studied using ever faster, and more complex machines. Often, the required processing power is delivered only by arguably the most complex systems of all — i.e., high performance parallel computers in their myriad of forms. To effectively exploit the available processing power, a thorough understanding of the complexity of such systems is required. As an immediate consequence, the number of 'passengers' that is capable of 'riding' such high performance parallel architectures is low.

Despite the complexity, many non-expert users are still tempted by the processing power provided by parallel systems — often to emerge with nothing but a disappointing result. In [75] this problem is stated somewhat more dramatically as follows:

> Anecdotal reports abound about researchers with scientific and engineering problems who have tried to make use of parallel processing systems, and who have been almost fatally frustrated in the attempt.

Clearly, there is a major discrepancy between the desire to obtain high performance with relative ease, and the potential of current high performance systems (i.e., the combination of all software layers and the underlying hardware) to satisfy this desire.

As indicated below, the specific research area of image processing — which is the field of focus of this thesis — also demonstrates a persistent desire to access the speed potential of high performance computer systems. The desire partially stems from the fact that it has been recognized for years that the application of parallelism in image processing can be highly beneficial [161]. However, in the field of image processing research, the observed discrepancy between desire and reality is no less severe. Essentially, the work described in this thesis is an endeavor to resolve this discrepancy — and to satisfy the need for easily obtainable speed in image processing.

1.1 The Need for Speed in Image Processing

The 'need for speed' has been recognized in many areas of digital image processing and computer vision [151]. Applications abound in which large amounts of data are to be processed, while having to adhere to strict time constraints at the same time. For example, a typical visual information standard such as television may generate data at a rate of up to 120 Mbytes per second [130]. As each pixel in the information stream generally is subjected to a multitude of processing steps, the total amount of processing power required per time unit is huge. In many cases (e.g., when real-time requirements are to be met), state-of-the-art sequential computers no longer can provide the necessary performance. The only way to supply the desired processing power (now and in the future) is by employing high performance computer systems.

A considerable diversity exists in the type of algorithms applied in imaging applications. Generally, a distinction is made between three different operation levels [118]:

1. **Low level image processing operations**. These operations primarily work on whole image data structures, and yield another image data structure. The computations have a local nature, and are to be performed for each pixel in an image. Examples are: basic filter operations (e.g., smoothing, edge enhancement), and image transformations (e.g., rotation, scaling).
2. **Intermediate level image processing operations**. These operations reduce the image data field into segments (regions of interest), and produce more compact and symbolic image structure representations (such as lists of object border descriptions). Examples are: region labeling, and object tracking.
3. **High level image processing operations**. These operations primarily concern the interpretation of the symbolic data structures obtained from the intermediate level operations. Essentially, the operations try to imitate human cognition and decision making, according to the information contained in the image. Examples are: object recognition, and semantic scene interpretation.

The execution of a set of low level routines is a common starting point for many typical image processing applications. In this thesis, we restrict ourselves to this initial processing phase. First, this is because the processing of visual data at the pixel level is highly regular in nature, to the effect that it provides a natural source of parallelism. More importantly, this is because the initial processing phase is by far the most time consuming part of the bulk of image processing applications [165].

1.2 The Gap Between Computing and Imaging

In spite of the large potential performance gains (and the overwhelming desire to obtain them), the image processing community at large does not benefit from high performance computing on a daily basis. As will be discussed extensively in this thesis, the problem is primarily due to the fact that no programming tool is available that can effectively help non-expert parallel programmers in the development of image processing applications for efficient execution on high performance computing architectures. Existing programming tools generally require the user to identify the available parallelism at a level of detail that is beyond the skills of non-expert parallel programmers [148]. As it is unrealistic to expect researchers in the field of image processing to become experts in high performance computing as well, it is essential to provide a tool that shields its users from *all* intrinsic complexities of parallelization.

The work described in this thesis is an attempt to effectively bridge the gap between the specific expertise of the image processing community, and the additional expertise required for efficient employment of high performance computer architectures. More specifically, the thesis describes the design and implementation of a *software architecture* that allows non-expert parallel programmers to develop image processing applications for execution on homogeneous distributed memory MIMD-style multicomputers. As a result, this thesis addresses the following fundamental research issue: how to design a sustainable, yet efficient, software architecture for parallel image processing, that provides the user with a *fully sequential* programming model, and hides all parallelization and optimization issues from the user completely.

1.3 Thesis Outline

In the past, several parallelization tools have been described that, to a certain extent, serve as an aid to non-expert parallel programmers. As discussed extensively in Chapter 2, such tools generally suffer from fundamental problems that make them unsuitable as an acceptable long-term solution for the image processing community. Most importantly, the tools often are provided with a programming model that does not match the image processing researcher's frame of reference. In addition, efficiency of parallel execution is often far from optimal. Also, it is often hard to incorporate extensions to deal with new hardware developments and additional user requirements. To overcome these problems, Chapter 2 proposes a new and innovative software architecture for *user transparent parallel image processing*, that excludes its users from having to learn *any* skills related to parallelization and performance optimization.

In Chapter 3, we give a detailed account of the software architecture's design philosophy. We focus on implementing the architecture such that code redundancy is avoided as much as possible, and efficiency of execution is guaranteed. We demonstrate that the presented design philosophy allows for long-term architecture sustainability, as well as close-to-optimal performance.

In Chapter 4, we indicate how to apply a simple analytical performance model in the process of automatic parallelization and optimization of complete image processing applications. To this end, we present a high level abstract parallel image processing machine (*APIPM*), designed to capture typical run-time behavior of parallel low level image operations. From its instruction set, a high level performance model is obtained that is applicable to a relevant class of parallel platforms.

Chapter 5 addresses the problem of accurate cost estimation of the communication primitives applied in our software architecture. It is observed that existing communication models are not powerful enough to serve as a basis for automatic and optimal domain decomposition of the image data structures applied in typical applications. To overcome this problem, the specific capabilities of the applied communication primitives are combined into a new, more powerful performance model (*P-3PC*).

Chapter 6 deals with the problem of the automatic conversion of any sequential image processing application into a correct and efficient parallel version. To this end, we define a simple finite state machine specification that guarantees the conversion process to be performed correctly at all times. As the issue of automatic optimization of complete applications is the central problem our software architecture for user transparent parallel image processing is confronted with, Chapter 6 combines all of the results obtained in Chapters 3, 4, and 5.

In Chapter 7, we give an assessment of the software architecture's effectiveness in providing significant performance gains. We describe the implementation and automatic parallelization of three well-known example applications that contain many operations commonly applied in image processing. In addition, we investigate how well the performance obtained with our software architecture compares to that of reasonable hand-coded implementations.

Finally, in Chapter 8 we summarize the results of this research, and present our view on the developed architecture for user transparent parallel image processing.