Summary

The goal of this thesis is to show a way to improve the performance of different radio astronomy applications. To begin with, in this thesis we advocate the use of many-core accelerators, parallel processors with hundreds of computational cores, as execution platforms for widely used radio astronomy algorithms and platforms. However, we also show that just using parallel hardware is not always enough to meet strict performance requirements. Therefore, to achieve real-time performance in the radio astronomy pipelines that are the use-cases of this thesis, we have to apply another fundamental optimization technique: auto-tuning. Auto-tuning is an optimization technique used to find the optimal configuration of a set of parameters, and in the context of this thesis we use it to find the best possible configurations of our parallels algorithms, on various many-core platforms, and for different use-case scenarios. In this thesis, by combining code generation with auto-tuning, we obtain code and performance portability for our applications, a result that is very important for a discipline like radio astronomy, where the life span of the instruments collecting data is much longer than the life span of the computers used to process these data.

In Chapters 3 and 4 we begin by showing how it is possible to improve the performance of two well-known radio astronomy algorithms, beam forming and dedispersion, by means of parallelization on many-core accelerators and auto-tuning. What we see for these two algorithms is that, both in terms of performance and energy efficiency, many-core accelerators provide better results than traditional multi-core CPUs. However, we also see that complex algorithms, running on platforms with such a high degree of parallelism, are difficult to configure and fine tune. We therefore demonstrate how auto-tuning is necessary to achieve high performance and performance portability.

In Chapters 5 and 6 we continue by showing that the combination of many-core accelerators and auto-tuning is not only beneficial for isolated algorithms, but also for more complex scientific pipelines. We do this by first looking at a prototype for the real-time pipeline of ARTS, the Apertif Radio Transient System, and then at a real-time pulsar detection pipeline, and conclude once again that

using many-core accelerators and auto-tuning it is possible to achieve real-time performance, a hard constraint for these scientific pipelines.

In Chapter 7 we conclude by showing how difficult, and at the same time how important, auto-tuning parallel applications running on many-core is. We are therefore able to generalize the importance of auto-tuning outside the domain of radio astronomy, and provide a quantitative definition of auto-tuning difficulty. We also show how this difficulty varies for different classes of algorithms, and for different platforms and input sizes.

To summarize, in this thesis we present experimental evidence that accelerating radio astronomy using many-cores and auto-tuning is a feasible and high-performance solution, and that this acceleration provides benefits that are both scientific and technological.