Space-Efficient Algorithms for Maximum Cardinality Search, Stack BFS, Queue BFS and Applications

Sankardeep Chakraborty[1](✉) and Srinivasa Rao Satti[2]

[1] The Institute of Mathematical Sciences,
CIT Campus, Taramani, Chennai 600113, India
sankardeep@imsc.res.in
[2] Seoul National University, 1 Gwanak-ro, Gwanak-gu, Seoul, South Korea
ssrao@cse.snu.ac.kr

Abstract. Following the recent trends of designing space efficient algorithms for fundamental algorithmic graph problems, we present several time-space tradeoffs for performing Maximum Cardinality Search (MCS), Stack Breadth First Search (Stack BFS), and Queue Breadth First Search (Queue BFS) on a given input graph. As applications of these results, we also provide space-efficient implementations for testing if a given undirected graph is chordal, reporting an independent set, and a proper coloring of a given chordal graph among others. Finally, we also show how two other seemingly different graph problems and their algorithms have surprising connection with MCS with respect to designing space efficient algorithms.

1 Introduction

Space efficient algorithms are becoming increasingly important owing to their applications in the presence of rapid growth of "big data". Another reason for the importance of space efficient algorithms is the proliferation of specialized handheld devices and embedded systems that have a limited supply of memory. As a consequence, algorithms that are oblivious to space constraint are not desired in such scenario. Even if mobile devices and embedded systems are designed with large supply of memory, it might be useful to restrict the number of write operations. For example, on flash memory, writing is a costly operation in terms of speed, and it also reduces the reliability and longevity of the memory. Keeping all these constraints in mind, it makes sense to consider algorithms that do not modify the input and use only a limited amount of work space. Such computational model has been proposed in algorithmic literature to study space efficient algorithms, and is known as the read-only memory (ROM) model. Following the recent trend, in this article, we focus on the space requirement for implementing some fundamental graph algorithms in such settings.

There is already a rich history of designing space efficient algorithms in the read-only memory model. In computational complexity theory, L is the complexity class [1] containing decision problems that can be solved by a deterministic

Y. Cao and J. Chen (Eds.): COCOON 2017, LNCS 10392, pp. 87–98, 2017.
DOI: 10.1007/978-3-319-62389-4_8