1 Verification of Graph Programs

Graphs are (discrete) mathematical structures used to represent relationships between objects [1]. They are often used to model structures and relationships so that they are related to many real problems. In computer science, graphs have been implemented in many areas of expertise. Graph theory can be applied in solving real problems by studying the rules that apply to graphs, including graph transformations. Graph transformation is an approach that modifies the graph structure by applying transformation rules. Perceptively, graph transformation is the process of changing a graph into a new shape by running a series of algorithms [2].

Many programming languages have been improved to facilitate the implementation of graph transformations, including the GP2 programming language. It is a rule-based, non-deterministic graph programming language that frees programmers from dealing with low-level graph data structures. The GP2 program algorithmically transforms the input graph into an output graph by sequentially applying the graph transformation rules. The results on the graph program can be formally verified using Hoare's logic [3]. There have been several manual proofs of the GP2 program by Poskitt and Wulandari [2,4,5,6]. Poskitt [4] uses a so-called E- and M-conditions as assertions to prove a graph program, while Wulandari [2] uses the monadic second-order logic (MSOL) for the proving. The E- and M-conditions use nested conditions as introduced in [7] where graphs isomorphism are considered in the assertions. Meanwhile, in MSOL introduced by Wulandari [2], the assertion used standard-like logic. Since the two types of proof methods are still done manually, human errors likely occur in the verification process.

On the other hand, Isabelle is a general proof assistant that enables mathematical formulas to be expressed in formal language and provides tools to prove those formulas in logical calculus. Isabelle's main application is the formalization of mathematical proof, especially formal verification which includes proving the validity of computer hardware or software as well as verifying the nature of personal computer languages and protocols [8]. Many studies use Isabelle/HOL, which is Isabelle's specialization for Higher-Order Logic. Isabelle/HOL was used for the extensive formalization of quantum algorithms and produced quantum information theory [4]. Isabelle/HOL can show how to prove the properties of functional programs by induction [9]. Previous exploration of HOL for formalizing graph theory has been conducted successfully [10]. The exploration in [10] considers twenty-one important definitions, six general definitions, twenty-eight auxiliary definitions, three pseudo definitions, and one theorem. The study also highlights the use of *tactic* in deriving a formal proof.

As mentioned previously, GP2 verification performed by Wulandari [2] uses standard logic, which can be defined in Isabelle. In this study, we present the use of Isabelle as a proof assistant to verify programming of graphs using the GP2 language. We chose Hoare Logic, Monadic Second Order Logic, and First Order Logic to help the verification of graph programming. To see how far Isabelle can be used as a proof assistant, we apply several case studies of graph programming using the GP2 programming language used in Wulandari's thesis. Several of these case studies were validated using Isabelle's proof assistant.