1 J − 3

An Evaluation of the Dependence-Based Complexity Metrics for Distributed Programs

Jianjun Zhao, Jingde Cheng, and Kazuo Ushijima
Dept. of Computer Science and Communication Engineering, Kyushu University

1. Introduction

Metrics for measuring software complexity have many applications in software engineering activities. Although a number of complexity metrics have been proposed and used for measuring sequential programs, few could be used for measuring concurrent and distributed software. Cheng proposed some dependence-based complexity metrics for distributed programs[1], but did not give the test and evaluation of these metrics. In order to optimize some rational and practical metrics for measuring the complexity of distributed programs, we selected various programs (27 programs) written in Occam 2 to test some of these metrics and evaluate each metric respectively according to the test results. This paper reports our test and evaluation results.

2. Program Dependences and Process Dependence Net

A *nondeterministic parallel control−flow net* (CFN) and a *nondeterministic parallel definition−use net* (DUN) are two program representations that can be used to represent multiple control flows and multiple data flows in distributed programs. A DUN can be regarded as a CFN with the information concerning definitions and uses of variables and communication channels[3].

Program dependences are dependence ralationship holding between statements in a program. Based on the DUN of a distributed program, we can formally define five kinds of primary program dependences, i.e.,control dependence, data dependence, selection dependence, synchronization dependence, and communication dependence. Because of the limitation of space, here we only give some informal descriptions for these dependences.

A statement u is directly control-dependent on the control predicate v of a conditional branch statement(e.g.,an if statement or while statement) if whether u is executed or not is directly determined by the evaluation result of v; a statement u is directly data-dependent on a statement v if the value of a variable computed at v has a direct influence on the value of a variable computed at u; a statement u is directly selection-dependent on a nonderministic selection statement v if whether u is executed or not is directly determined by the selection result of v; a statement u is directly synchronization-dependent on another statement v if the start and/or termination of execution of v directly determines whether or not the execution of u starts and/or terminates; a statement u in a process is directly communication-dependent on another statement v in another process if the value of a variable computed at v has direct influence on the value of a variable computed at u by an interprocess communication.

If we represent all five kinds of primary program dependences in a distributed program within an arc-classified digraph such that each type of arcs represents a kind of primary program dependences, then we can obtain an explicit dependence-based representation of the program. Cheng named such a representation the "Process Dependence Net" [2,3].

The *process dependence net* (PDN) of a program is an arc-classified digraph $(V, Con, Sel, Dat, Syn, Com)$, where V is the vertex set of the CFN of the program, Con is the set of control dependence arcs such that any $(u,v) \in Con$ iff u is directly control-dependent on v, Sel is the set of selection dependence arcs such that any $(u,v) \in Sel$ iff u is directly selection-dependent on v, Dat is the set of data dependence arcs such that any

$(u, v) \in Dat$ iff u is directly data-dependent on v, Syn is the set of synchronization dependence arcs such that any $(u, v) \in Syn$ iff u is directly synchronization-dependent on v, and Com is the set of communication dependence arcs such that any $(u, v) \in Com$ iff u is directly communication-dependent on v [2,3].

3. Dependence-Based Complexity Metrics

Below, we will use the following notations of relational algebra:
$\sigma_{[1]=v}(R)$: the selection of binary relation R such that $\sigma_{[1]=v}(R) = \{(v1, v2) | (v1, v2) \in R \text{ and } v1 = v\}$.
$\sigma_{[1]\in s}(R)$: the selection of binary relation R such that $\sigma_{[1]\in s}(R) = \{(v1, v2) | (v1, v2) \in R \text{ and } v1 \in S\}$. $Dp \in \{Con, Sel, Dat, Syn, Com\}$, $Dpc \in \{Sel, Syn, Com\}$, $Du = Con \cup Sel \cup Dat \cup Syn \cup Com$, P is the set of all statements of a process named P. $|A|$ is the cardinality of set A.

The test results and the evaluation of each metric are as follow:

(a). $|Dp|/|Du|$: This is the proportion of a special primary program dependence to all primary program dependences in a program. The test results show: this metric can only be used to measure the degree of concurrency of a program from a special viewpoint.

(b). $|Sel \cup Syn \cup Com|/|Du|$: This is the proportion of those primary program dependences concerning concurrency to all primary program dependences in a program. The test results show: this metric can be used to measure the degree of concurrency of the program from a general viewpoint.

(c). $\max/\min\{\,|\sigma_{[1]=v}(D_p)|\,\big|\, v \in V\}/|V|$: These are the proportions of the maximal(minimal) number of statements, on which a statement is directly control-, data-, selection-, synchronization-, or communication-dependent, respectively, to the total number of statements in a program. The test results show: this metric is neither rational nor practical metric.

(d). $\max/\min\{\,|\sigma_{[1]=v}(D_{pc})|\,\big|\, v \in V\}/|V|$: These are the proportions of the maximal(minimal) number of statements, on which a statement is directly selection-, synchronization-, or communition- dependent, respectively, to the total number of statements in a program. The test result show: this metric is neither rational nor practical metric.

(e). $\max/\min\{\,|\sigma_{[1]} = v(D_u)|\,\big|\, v \in V\}/|V|$: These are the proportion of the maximal(minimal) number of statements, on which a statement is somehow directly dependent, to the total number of statements in a program. The test results show: this metric is neither rational nor practical metric.

(f). $|\sigma_{[1]\in P}(Syn \cup Com)|/|V|$: This is the proportion of the number of statements of other processes, on which a process is directly dependent, to the total number of statements in a program. The test results show: this metric can be used to measure the complexity of the concurrency in a program.

4. Conclusion

We used 27 programs written in Occam 2 to test the dependence-based complexity metrics for distributed programs proposed by Cheng. From the test results, we can conclude that: *metric*(a) can be used to measure the degree of concurrency of a distributed program from a special viewpoint; *metric*(b) can be used to measure the degree of concurrency of a distributed program from a general viewpoint; *metrics*$(c), (d),$ *and*(e) are neither rational nor practical met ics; *metric*(f) can be used to measure the complexity of concurrency of a distributed program.

References

[1] J. Cheng, "Dependence-Based Complexity Metrics for Distributed Programs," to appear in *Proceedings of the 22nd Annual International Conference on Parallel Processing*, St.Charles, U.S.A., August 16-20, 1993.

[2] J. Cheng, "Process Dependence Net: A Concurrent Program Representation," *Proc. JSSST 8th Conference*, pp.513-516, September 1991.

[3] J. Cheng, "Process Dependence Net of Distributed Programs and Its Applications in Development of Distributed Systems," to appear in *Proc.IEEE-CS 17th Annual COMPSAC*, Phoenix, U.S.A., November 1-5, 1993.