Abstract

Instruction-level parallelism (ILP) is a family of processor and compiler design techniques that speed up the execution of sequential programs by causing individual machine operations to execute in parallel.

Execution speed of sequential programs on ILP architectures is sensitive to the order in which instructions are presented to the processor. Dependencies between instructions (data dependencies, control dependencies and resource conflicts) present a major hurdle to the amount of instruction-level parallelism. To realize potential execution efficiency and reduce or avoid hazard dependencies instruction stream requires to be ordered such that, whenever possible, independent multiple low-level operations can be executed in parallel. This reordering of machine operations is typically called *instruction scheduling* which becomes one of the last phases performed by modern optimizing compilers. This scheduling problem has been shown to be NP-complete, i.e., the time required to compute an optimal schedule increases exponentially with the size of the problem.

This work describes a genetic algorithm approach to the instruction scheduling problem called Genetic Algorithm Based Scheduler (GABS) in the presence of data dependencies and resource constraints only, i.e., local instruction scheduling. So that, ordering instructions within a basic block such that the schedule length is minimized without violating precedence constraints (dependence constraints). The genetic algorithm, rather than pure one, is improved with the introduction of some knowledge about the scheduling problem represented by the use of heuristic priorities, as repair method, assigned at each instruction in order to respect precedence constraints and prune the expensive search space caused by the infeasible schedules. Two repair methods are implemented to correct any unfeasible schedule generated by genetic operators. To test the efficiency of this system (GABS), some basic blocks are evaluated and the results are compared in terms of schedule length to those obtained using list scheduling heuristics. Performance results from simulations demonstrate the efficiency of the proposed algorithm to found solutions in most cases after just a little number of generations.

PDF created with pdfFactory trial version www.pdffactory.com

الملخص

إن المعالجات التي تقوم بتنفيذ مجموعة من الإيعازات على التوازي وفي وقت واحد (Parallelism Instruction-Level) مع المترجمات (Compilers) الملحقة بها توفر إمكانية زيادة سرعة تنفيذ البرامج التعاقبية بسبب تنفيذ أكثر من إيعاز في وقت واحد.

هذا النوع من المعالجات يتأثر جدا بتسلسل الإيعازات في البرنامج المطلوب تنفيذه. فالعلاقة أو الإعتمادية بين الإيعازات سواءً كانت بسبب توفير المعطيات ، ترابط السيطرة، او التنافس على المعطيات تمثل أكبر عائق تجاه تنفيذ هذه الإيعازات بشكل متوازي على هذا الاساس فإن تسلسل أو ترتيب الإيعازات في البرنامج الواحد مهم جدا في تجاوز مشكلة الإعتمادية أعلاه ومن هنا تبرز مشكلة البحث حول أفضل ترتيب للتنفيذ يمكن إتباعه لكل برنامج. إنّ الزمن المطلوب لعملية حساب أفضل ترتيب أو جدولة للإيعازات (Instruction scheduling) يتناسب طرديا و بشكل أسي مع حجم البرنامج أو المشكلة.

هذا العمل يقدم توصيف المشكلة أعلاه باستخدام الخوارزميات الجينية (Genetic algorithms) ممّا يوفر طريقة لإيجاد افضل جدولة وسميت هذه الطريقة بـ (GABS)، تعمل هذه الطريقة مع وجود المحددات الخاصة بالإعتمادية بين الإيعازات بسبب المعطيات أو بسبب التنافس على مصادرها ضمن المقطع الواحد من البرنامج (Basic block) ، و عليه فإنّ جدولة الإيعازات تحصل موقعيًا (local) لكل مقطع في البرنامج.
لغرض تحسين أداء الخوارزميات الجينية في حل هذا النوع من المشاكل، تمّ تعزيزها ببعض المحددات عن أولويات التنفيذ الاستكشافي (Heuristic priorities) التي تمّ تحديدها لكل ايعاز بما يوفر إمكانية وضع أسبقيات التنفيذ للإيعازات. ذلك ممّا يؤدي الى تشذيب أو اختزال مساحة البحث الواسعة الناتجة عن احتمالات الجدولة غير المفيدة.

لأجل فحص كفاءة الطريقة المقترحة (GABS) تمّ تطبيقها على عدد من مقاطع البرامج و إستحصال النتائج و مقارنتها مع طريقة قوائم الجدولة الاستكشافية (List scheduling heuristics) حيث اعتمدت المقارنة على مبدأ طول فترة التنفيذ لكل مقطع في البرنامج، ولقد أوضحت النتائج إقتدار الخوارزمية المقترحة في أيجاد حلول (near-optimal solutions) وذلك في معظم الحالات و في زمن مستحسن.

PDF created with pdfFactory trial version www.pdffactory.com