Parallelizing dynamic sequential programs using polyhedral process networks

Nadezhkin, D.

Samenvatting

Dit proefschrift concentreert zich op de automatische parallellisatie van sequentiele programma's met een zodanig adaptief en dynamisch gedrag dat zij geexecuteerd kunnen worden op een embedded systeem met meerdere processoren. In ons werk leiden wij Polyhedrale Proces Netwerken (PPN) specificaties af uit dynamische sequentiele programma's. Het sequentiele model matcht niet met de manier waarop multiprocessor systemen werken. Het PPN model is een deelverzameling van het Kahn Process Netwerk model, en staat veel dichter bij multiprocessor systemen. PPN kan met wiskundige methoden benaderd worden.

De methoden en technieken van dit proefschrift zijn belangrijke en niet-triviale uitbreidingen op voorafgaand werk over systematische en automatische afleiding van parallelle specificaties in de vorm van procesnetwerken, uit programma's die bestaan uit statische programmeerconstructies ("loops"). Deze programma's staan een automatische analyse toe en kunnen direct vertaald worden naar PPN's. De restrictie tot statische programma's geeft echter beperkte toepasbaarheid, programma's met adaptief en dynamisch gedrag passen niet binnen deze restrictie. Daarom proberen we in dit proefschrift deze beperkingen te verminderen.

Door het bestuderen van verschillende toepassingen besloten wij de volgende uitbreidingen van programma's te bekijken:

(1) dynamische if-condities worden toegestaan

(2) for-loops met dynamische grenzen worden toegestaan

(3) while-loops worden toegestaan.

Voor **(1)** hebben we een nieuwe methode ontwikkeld die werkt voor een klasse van affine loopprogramma's met dynamische if-condities. Deze condities kunnen afhankelijk zijn van informatie die niet bekend is op het tijdstip van de compilatie and die gedurende run-time aan verandering onderhevig kan zijn. We hebben ook gekeken naar de moeilijkere uitbreidingen **(2)** en **(3)**. In hoofdstuk 3 is een eerste stap gezet voor programma's met affine nested loops met dynamische grenzen. In

hoofdstuk 4, geven wij een nieuwe aanpak voor affine nested-loop programma's met while-loops.

In tegenstelling tot het afleiden van een PPN specificatie uit een statisch programma, is het systematisch en automatisch omzetten van dynamische programma's naar PPN's een complex en uitdagend probleem. Dit komt omdat de belangrijke stappen in het afleiden van een PPN programma dan niet werken. In hoofdstukken 3 en 4 laten we zien dat, ondanks dat het precieze gedrag van dynamische programma's niet bekend is op het tijdstip van de compilatie, het toch mogelijk is om dergelijke programma's te analyseren en te transformeren naar executeerbare PPN specificaties op een systematische en automatische manier.

In hoofdstuk 5 wordt een andere contributie beschreven. The PPN model gebruikt FIFO kanalen als communicatiemedium. Het gevolg is dat toegang tot het geheugen in het sequentiele programma geconverteerd moet worden naar dataflow over de FIFO kanalen. Hoofdstuk 5 laat zien hoe dit gedaan kan worden: communicatie karakteristieken tussen twee communicerende processen in PPN worden bestudeerd.

De methoden en technieken die in dit proefschrift gepresenteerd worden, hebben geresulteerd in uitbreidingen van de pn compiler (http://daedalus.liacs.nl). In hoofdstuk 6 wordt de parallellisatieaanpak van hoofdstuk 3 geevalueerd op de toepassing "Low Speed Obstacle Detection".