KOMPUTASI PARALEL ASINKRON PADA JARINGAN SARAF TIRUAN

Agus Virgono
Departemen Teknik Elektro Prodi Teknik Komputer Institut Teknologi Telkom
Jl. Telekomunikasi 1, Dayeuh Kolot Bandung 40257
Telepon Kantor : 022-7565933, Telpon Pribadi : 022-76065979
Email : agv@stttelkom.ac.id

ABSTRAK

Jaringan Saraf Tiruan (JST) merupakan suatu model matematis yang mengambil sifat-sifat komputasi Jaringan Saraf Manusia. Selama ini untuk memudahkan pemodelan, pada JST waktu proses tiap sel saraf manusia yang disebut neuron dianggap sama/sinkron (Synchronous Paralel Processing/SPP). Penelitian ini menganalisa JST bila waktu proses tiap neuron tidak sama (Asynchronous Paralel Processing/APP) dan membandingkan performansi antara JST APP dan JST SPP. Sebagai studi kasus, diambil salah satu algoritma JST yaitu Propagasi Balik (Back Propagation/BP).

Simulator digunakan untuk menghasilkan data rata-rata jumlah kuadrat error (Eav) dan jumlah pola uji yang dikenali dengan benar. Dari data Eav tersebut dapat dihasilkan grafik Eav, dan dari jumlah pola uji yang dikenali dengan benar dapat ditentukan Tingkat Generalisasi (TG). Data dan grafik Eav serta Tingkat Generalisasi dijadikan parameter performansi.

Setelah dilakukan penelitian diperoleh hasil bahwa pada beberapa jaringan JST Propagasi Balik APP (JST BP APP) memiliki performansi lebih baik dibanding JST Propagasi Balik SPP (JST BP SPP). Hal ini membuka kemungkinan untuk memperoleh bobot yang lebih baik pada suatu aplikasi yang telah memperoleh bobot terbaik hasil JST BP SPP.

1. PENDAHULUAN

Penelitian ini bertujuan untuk mengamati proses komputasi paralel JST (umumnya dilakukan secara serial di sebuah komputer sehingga hasil dari komputasi setiap sel JST selalu sinkron) jika digunakan metoda asinkron dimana setiap sel akan mengeluarkan hasil sesuai dengan kecepatan komputasinya sendiri-sendiri, sehingga akan didapatkan proses komputasi yang asinkron.

2. LANDASAN TEORI

Ditinjau dari sudut bahasa, paralel adalah kemampuan melakukan beberapa kegiatan dalam satu waktu. Sedangkan beberapa defenisi proses paralel antara lain adalah melakukan beberapa kegiatan berbeda dalam satu waktu, membagi satu kegiatan menjadi beberapa bagian yang bekerja secara bersamaan, penggunaan k *resource* (sumberdaya) untuk menyelesaikan n pekerjaan, penggunaan k *resource* untuk menyelesaikan satu pekerjaan.

Beberapa kendala timbul akibat mekanisme yang harus dijalani oleh proses paralel. Idealnya bila suatu aplikasi dijalankan pada n buah prosesor *uniform* (seragam) secara paralel, maka kecepatan komputasi untuk aplikasi tersebut akan meningkat sebesar n kali. Namun keadaan ini jarang dicapai karena beberapa hal berikut :

- Sangat sulit menjaga tiap prosesor berjalan secara terus menerus pada kecepatan maksimumnya.
- Ketergantungan data suatu prosesor terhadap prosesor lainnya.
- Sebuah prosesor harus menunggu data hasil dari prosesor lain sebelum menjalankan prosesnya.
- Pemakaian *recourse* (sumber daya) secara bersama.
- Biasanya prosesor-prosesor yang bekerja secara paralel menggunakan *recourse-recourse* secara bergantian dimana pada satu saat hanya boleh ada satu prosesor yang menggunakan satu *recourse*. Hal ini menyebabkan sebuah prosesor harus menunggu sebuah sumber daya yang akan dipakai bila sumber daya tersebut sedang dipakai oleh prosesor lain.

JST merupakan model matematis yang mengambil inspirasi dari struktur dan sistem kerja JSM yang mempunyai ciri [3]:

- Terdiri dari sejumlah besar elemen proses sederhana yang disebut *neuron*.
- Antar *neuron* dihubungkan oleh *synapses* yang berisi bobot sebagai dasar pengetahuan
- Bekerja secara *massive parallelism*
- Mampu belajar dari pengalaman

Secara garis besar pada JST terdapat dua tahap komputasi yaitu :

- Tahap Belajar
  Pada tahap ini proses dimulai dengan memasukkan pola-pola belajar kedalam jaringan. Dengan menggunakan pola-pola ini jaringan akan mengubah-ubah bobot yang menjadi penghubung antara *node*. Satu periode

dimana seluruh pola belajar telah diproses yang disebut 1 (satu) iterasi. Pada setiap 1 iterasi dilakukan evaluasi terhadap keluaran jaringan. Tahap ini berlangsung pada beberapa iterasi dan berhenti setelah jaringan menemukan bobot yang sesuai dimana suatu keadaan yang diinginkan telah terpenuhi. Selanjutnya bobot ini menjadi *knowledge base* (dasar pengetahuan) pada tahap pengenalan.

- Tahap Pengenalan
  Pada tahap ini dilakukan pengenalan terhadap suatu pola masukan dengan menggunakan bobot hasil tahap belajar.

Dalam memproses informasi ditinjau dari ada tidaknya umpan balik, secara garis besar terdapat dua arsitektur JST, yaitu :

- Arsitektur Umpan Maju (*Feed forward*)
  Aliran informasi masuk melalui lapisan *input* diteruskan ke lapisan *hidden* dan selanjutnya ke lapisan *Output.* Pada lapisan *hidden* dan *Output* dilakukan pengolahan informasi tanpa umpan balik.
- Arsitektur Balik Maju (*Back forward*)
  Pada arsitektur ini pengolahan informasi berlawanan dengan arsitektur umpan maju dimana keluaran dari suatu *node* diumpan balikkan menjadi nilai masukan bagi *node* lain.

JST Propagasi Balik APP

Perbedaan waktu proses tiap *node* dalam menghitung keluaran, menyebabkan perbedaan keluaran yang digunakan JST BP APP dan JST BP SPP. Bila pada JST BP SPP keluaran yang digunakan selalu yang terbaru, maka pada JST BP APP keluaran terbaru dipakai bila sebuah *node* telah menghasilkan, sedangkan untuk yang belum digunakan keluaran lama.

Proses yang terjadi pada tahap belajar dijelaskan dengan mengamati aliran informasi selama *two past computation.*

- *Forward pass*
  Pada langkah ini dilakukan perhitungan keluaran lapisan demi lapisan. Proses pada suatu lapisan akan dimulai bila Jumlah *Node* Selesai (JNS) suatu lapisan telah memenuhi. Bila t adalah saat perhitungan keluaran suatu lapisan dimulai, maka keluaran suatu *node* pada lapisan tersebut dirumuskan dengan

$$y^{h}_{(a+1)j} = f_{sigmoid}\left(\sum_{i=1}^{N}\left(C_{(t)i} w^{h}_{ij}\right) + \phi^{h}_{j}\right) \quad .....(2.1)$$

$$y^{o}_{(a+1)k} = f_{sigmoid}\left(\sum_{j=1}^{N}\left(y^{h}_{(t)j} w^{o}_{jk}\right) + \phi^{o}_{k}\right) \quad (2.2)$$

Setelah dilakukan perhitungan keluaran, selanjutnya dilakukan perhitungan *error*

$$e_k(n) = d_k(n) - y^{o}_{(t)k}(n) \qquad (2.3)$$

- *Backward pass*

Pada JSM perubahan bobot disebabkan oleh adanya sinyal keluaran dari suatu *neuron* ke *neuron* lain. Hal ini menunjukkan bahwa komputasi perubahan bobot pada JST keluaran suatu *node* digunakan sama dengan keluaran yang digunakan pada perhitungan keluaran. hal ini menyebabkan gradien *error* lokal yang dirumuskan dengan

$$EO_k = y^{o}_{(t)k}.(1 - y^{o}_{(t)k}).e_k \qquad (2.4)$$

$$EH_j = y^{h}_{(t)j}.(1 - y^{h}_{(t)j}).\sum_{k=1}^{Q} E_k w_{kj} \quad (2.5)$$

dan perubahan bobot dirumuskan dengan

$$w^{o}_{(a+1)jk} = w^{o}_{(a)jk} + \alpha.EO_k.y^{h}_{(t)j} \quad (2.6)$$

$$w^{h}_{(a+1)ij} = w^{h}_{(a)ij} + \alpha.EH_j.C_{(t)i} \qquad (2.7)$$

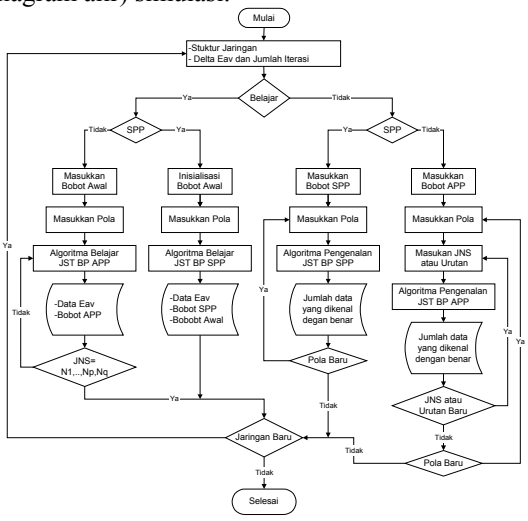
Nilai $C_{(t)i}$, $y^{h}_{(t)j}$, $y^{o}_{(t)k}$ sama dengan $C_{(a+1)i}$, $y^{h}_{(a+1)j}$, $y_{(a+1)k}$ untuk *node-node* yang berstatus 1 dan $C_{(a)i}$, $y^{h}_{(a)}$, $y^{o}_{(a)k}$ untuk yang berstatus 0.

Metoda Batas Pemberhentian Tahap Belajar

Secara matematis makna dari keluaran *node* berstatus 0 adalah *node* tersebut dihubungkan dengan *node* pada lapisan sebelumnya melalui bobot lama atau bobot pada pola ke a (w_a) dan *node* berstatus 1 dihubungkan melalui bobot terbaru atau pada pola ke a+1 (w_{a+1}). Sedangkan untuk menghitung nilai E(n) yang digunakan adalah w_{a+1}. Hal ini menyebabkan seluruh perhitungan untuk menghasilkan E(n) (perhitungan keluaran dan error) harus menggunakan nilai terbaru atau dengan kata lain dilakukan secara SPP.

3. PERANCANGAN SIMULATOR

Gambar dibawah ini memperlihatkan *flow chart* (diagram alir) simulasi.



Gambar 3.1 *Flow chart* Simulasi

Secara garis besar simulasi ini dibagi kedalam 5 langkah yaitu Masukan Simulasi, Bobot Awal, Pola Masukan, Algoitma dan Hasil Simulasi.

a. Masukan Simulasi
   Pada langkah ini dilakukan pembentukan struktur jaringan. Khusus pada tahap belajar dimasukkan ΔEav dan jumlah iterasi.
b. Bobot Awal
   Pada langkah ini dimasukkan bobot yang akan digunakan pada algoritma JST BP. Bobot yang digunakan tergantung pada tahap komputasi (belajar atau pengenalan) dan jenis proses paralel (SPP atau APP). Pada tahap belajar digunakan bobot awal dan pada tahap pengenalan digunakan bobot hasil belajar jaringan tersebut. Bobot awal berasal dari inisialisai bobot bila jenis proses paralelnya SPP, sedangkan bila APP digunakan bobot awal yang dihasilkan pada simulasi tahap belajar JST BP SPP untuk jaringan yang sama. Tahap komputasi dan jenis proses paralel ditentukan setelah langkah masukan simulasi dan sebelum langkah bobot awal.
c. Pola Masukan
   Pada langkah ini dimasukkan pola yang akan digunakan pada tahap belajar atau pengenalan.
d. Algoritma JST BP
   Pada proses ini dijalankan algoritma JST BP sesuai dengan tahap komputasi dan jenis proses paralelnya. Algoritma pengenalan dapat diulang-ulang untuk pola yang berbeda. Khusus untuk JST BP APP pada tahap belajar dilakukan berulang-ulang dari JNS 1,..,1,1 sampai N_1,..,N_P,N_Q. Sedangkan pada tahap pengenalan untuk pola yang sama dapat diulang pada JNS yang berbeda.
e. Hasil Simulasi
   Pada langkah ini dilakukan penyimpanan keluaran simulasi. Keluaran simulasi ini digunakan sebagai data yang dianalisa.

4. ANALISA DATA

Percobaan dan analisa dilakukan pada tahap belajar dan pengenalan JST Propagasi Balik SPP dan APP. Beberapa bentuk jaringan yang ditentukan secara sembarang, pada tahap belajar dilatih mengenali 10 pola belajar dan pada tahap pengenalan di uji mengenali 50 pola uji. Pola belajar dan pola uji terdapat pada pola percobaan dalam lampiran B halaman 1-3. Setiap jaringan menggunakan konstanta belajar sebesar 0.35 [3], Δe_{av} 10^{-6},10000 iterasi dan toleransi *error* (ε) sebesar 0,5 (optimalisasi pengenalan).

Bentuk-bentuk jaringan yang digunakan adalah 75, 85, 36, 56, 37, 57 untuk jaringan dengan 1 lapisan *hidden* dan 345, 365, 645, 536, 846, 347, 467 untuk 2 lapisan *hidden*.
Setiap satu pola *input* dipasangkan dengan satu pola *output*. Untuk itu pada percobaan ini digunakan sebanyak 3 jenis pola *output* yang masing-masing terdiri atas 10 pola.

1 Uji Simulator

Uji simulator dilakukan untuk mengetahui kebenaran hasil simulasi. Uji simulator dapat dilakukan dengan cara membandingkan hasil yang diperoleh dengan keadaan yang diinginkan. Pada JST BP keadaan yang diinginkan adalah kemampuan mengenali seluruh pola belajar. Sebagai contoh diambil salah satu JST BP SPP jaringan 345. Pada tahap belajar diperoleh hasil sebagai berikut.

Tabel 4.1 Hasil tahap belajar JST BP SPP 345

Eav terkecil	Iterasi	Eav konvergen	Iterasi
0,0006389	10000	0,0006389	10000

Nilai Eav konvergen yang cukup kecil pada tabel 4.1 menunjukkan bobot yang dihasilkan jaringan tersebut cukup baik.

JST Propagasi Balik APP

Dengan mengunakan bobot awal dan kondisi jaringan yang digunakan pada tahap belajar JST BP SPP, dilakukan tahap-tahap percobaan sebagai berikut.

- Seluruh jaringan dicoba satu kali.
- Setiap JNS yang memiliki nilai Eav terkecil <1 dicoba sebanyak 4 kali. Hal ini dilakukan dengan harapan dihasilkan nilai yang lebih baik.
- Setiap JNS yang memiliki nilai Eav terkecil $<10^{-1}$ dicoba sebanyak 25 kali.
  Hal ini dilakukan dengan harapan dihasilkan nilai yang lebih baik.

Dari 3 tahap percobaan tersebut telah dilakukan sebanyak 30 kali percobaan. Jumlah percobaan ini dianggap cukup untuk melihat perbandingan performansi. Dari data-data yang dihasilkan diperoleh beberapa data yang memiliki nilai Eav terkecil dan Eav konvergen yang lebih baik dari JST BP SPP. Pada tabel 4.2 diperlihatkan 1 buah data dari beberapa data tiap jaringan yang memiliki performansi tahap belajar JST BP APP lebih baik dari JST BP SPP .

Tabel 4.2 Data hasil tahap belajar JST BP APP yang lebih baik dari JST BP SPP

No	Jari-ngan	JNS	Eav terkecil	Iterasi	Eav konvergen	Iterasi
1	7 5	SPP	0.0001607	10000	0.0001607	10000
		6 3	0.0000865	7020	0.0000973	7018
		6 4	0.0000184	7411	0.0000184	7411
		6 5	0.0000821	8388	0.0000945	8394
		7 3	0.0000570	9312	0.0000570	9312
2	8 5	SPP	0.0001442	10000	0.0001442	10000
		7 3	0.0000028	6699	0.0000028	6699
		7 4	0.0000219	9777	0.0000982	9779
		7 5	0.0000961	9972	0.0000970	9690
3	5 6	SPP	0.0002115	10000	0.0002115	10000
		5 5	0.0001464	7682	0.0001614	7681
4	5 7	SPP	0.0002016	10000	0.0002016	10000
		5 6	0.0001561	8072	0.0001652	9630
5	8 4 6	SPP	0.0006202	10000	0.0006202	10000
		7 4 6	0.0005144	9680	0.0005375	8494

Dari tabel 4.2 terlihat bahwa untuk setiap jaringan selalu terdapat JNS kurang 1 *node* dari jumlah *node* pada salah satu lapisannnya (misal JNS 65 untuk jaringan 75).

Pengenalan pola belajar JST BP APP

Pada percobaan pengenalan pola belajar ada jaringan yang mampu mengenali seluruh pola dan ada yang tidak. Setiap jaringan yang mampu mengenali seluruh pola belajar selanjutnya dicoba untuk mengenali pola uji. Bila data tersebut dibandingkan dengan hasil JST BP SPP maka terdapat beberapa data yang lebih baik. Data-data tersebut sebagai berikut.

Tabel 4.3 Data pengenalan pola uji JST BP APP yang lebih baik dari JST BP SPP

No	Jar	JNS	Kenal										Jlh
			40%	42%	48%	50%	52%	54%	56%	58%	60%	62%	
1	7 5	SPP						1					1
		64							1				1
		73							2	1			3
2	8 5	SPP							1				1
		73								1		1	2
3	3 6	SPP											1
		35			10	2	1						13
4	5 6	SPP					1						1
		45						1	1				2
5	3 6 5	SPP				1							1
		355									1		1
6	6 4 5	SPP					1						1
		545						1	1		1		3
		644						4		3	1	2	13
7	5 3 6	SPP	1										1
		436		3									3
		535					2	9	5	5		2	23

Angka-angka pada kolom kenal dan kolom jumlah menunjukkan jumlah percobaan. Misal pada jaringan 36 dengan JNS 35, dari 25 kali percobaan diperoleh data yang lebih baik sebanyak 13 percobaan yang terdiri dari mampu mengenali 48% sebanyak 10 percobaan, 50 % 2 percobaan dan 52% 1 percobaan.

Pada tabel 4.3 terlihat data hasil pengenalan pola uji JST BP APP yang lebih baik dari JST BP SPP. Dari 25 kali percobaan kemungkinan mendapatkan data yang lebih baik bervariasi untuk tiap jaringan. Namun kemungkinan terbesar pada jaringan dengan 2 lapisan *hidden* selalu diperoleh jaringan dengan JNS kurang 1 *node* dari jumlah *node* pada salah satu lapisannnya. Misal JNS 538, 448 dan 547 untuk jaringan 548. Sedangkan pada jaringan dengan 1 lapisan *hidden* dihasilkan jaringan dengan JNS lapisan *hidden* minimal kurang 1 *node* dari jumlah *node* nya. Misal JNS 81, 82, .., 86, 91, 92, …, 94 dan 95 untuk jaringan 96.

5. KESIMPULAN DAN SARAN

5.1 Kesimpulan

a. JST BP APP lebih baik dari JST BP SPP pada tahap pengenalan, sedangkan JST BP SOO lebih baik pada tahap belajar

b. Pengenalan pola uji menunjukkan bahwa data JST BP APP yang lebih baik dari JST BP SPP dihasilkan bila.
   - Jaringan 1 lapisan hidden
     JNS lapisan hidden minimal kurang 1 node dari jumlah node lapisan *hidden*. Misal JNS 81, 82, .., 86, 91, 92, …, 94 dan 95 untuk jaringan 96.
   - Jaringan 2 lapisan hidden
     JNS pada salah satu lapisan kurang 1 node dari jumlah node nya. Misal JNS 538, 448 dan 547 untuk jaringan 548

5.2 Saran

a. Bila bobot terbaik JST BP SPP suatu aplikasi yang memiliki 1 atau 2 lapisan *hidden* telah ditemukan, sebaiknya dilakukan 25 kali tahap belajar dan pengenalan JST BP APP pada JNS tertentu ..

b. Pada penelitian selanjutnya perlu dilakukan penelitian tentang :
   - JST BP APP untuk jaringan dengan 3, 4 atau 5 lapisan hidden sesuai dengan struktur jaringan sering digunakan.
   - APP pada algoritma JST lain yang sudah ada.
   - Algoritma JST baru yang dapat menerapkan APP JSM secara murni.

REFERENSI

[1] Desiani, A. dan Arhami, M. *Konsep Kecerdasan Buatan*. ANDI Yogyakarta, Yogyakarta, 2006.

[2] Hermawan, A. *Jaringan Syaraf tiruan Teori dan Aplikasi.* ANDI yogyakarta, Yogyakarta, 2006.

[3] Myers, Catherine E., *Delay Learning in Artificial Neural Network.* Chapman & Hall London, London 1992.

[4] Neilsen, F. *Neural Networks – algorithms and applications*. Niels Brock Business College, 2001.

[5] Puspitaningrum, D. *Pengantar Jaringan Syaraf Tiruan*. ANDI Yogyakarta, Yogyakarta, 2006.

[6] Siang, J.J., *Jaringan Saraf Tiruan dan Pemogramannya Menggunakan Matlab*, ANDI Yogyakarta, Yogyakarta 2005.

ISSN: 1907-5022