APLIKASI METODE *NEURO-DYNAMIC* PADA PROSES PENGENDALIAN PERSEDIAAN DI SEBUAH PERUSAHAAN RETAIL

Ngarap Im Manik
FMIPA, Jurusan Matematika, Universitas Bina Nusantara
Jl. Kebon Jeruk Raya No. 27 Kemanggisan / Palmerah Jakarta Barat
Email: manik@binus.edu

ABSTRAK

Masalah yang sering dihadapi perusahaan retail saat ini antara lain adalah sistem pengendalian persediaan yang kurang tepat, terutama ditinjau dari segi biaya dan kepuasan pelanggan. Hal ini disebabkan karena kuantitas pemesanan yang ditetapkan perusahaan terlalu besar maupun terlalu kecil, sehingga mengakibatkan barang yang disimpan tidak optimal. Untuk menyelesaikan masalah di atas diusulkan dengan menggunakan metode *Neuro-Dynamic*, yang dapat mengoptimalkan stok barang yang tersedia pada toko, agar biaya yang dikeluarkan minimal dengan menghasilkan keuntungan yang sebesar-besarnya. Untuk memudahkan proses perhitungannya dilakukan dengan program komputer. Hasil pembahasan menunjukkan bahwa perusahaan retail tersebut telah terbantu dalam menjalankan aktivitas perdagangannya, karena tersedianya informasi yang dibutuhkan secara cepat dan cukup akurat dan dapat mengoptimalkan jumlah stok barang yang ada, sehingga mengoptimalkan *overdemand* dan *oversupply*.

Kata kunci: pengendalian persediaan, *Neuro-Dynamic*, optimasi.

ABSTRACT

Improper inventory system in terms of cost and customer satisfaction is always the most common problem in every part of industries, and of course, also for retailers. The inappropriateness in setting of orders makes the warehouse overloaded for some kind of goods and lack for the others. To overcome this problem, we proposed to apply the Neuro-Dynamic method for optimizing the composition of goods. This method can be easily implemented using computer program and the result indicates that the overdemand and oversupply problems in that company are solved.

Keywords: *inventory control, Neuro-Dynamic, optimize.*

1. PENDAHULUAN

Untuk mempermudah operasional perusahaan dalam mencapai keuntungan sebesar-besarnya dengan ongkos seminimal mungkin, maka saat ini banyak perusahaan telah dan akan memanfaatkan teknologi informasi sebagai alat bantu dalam proses bisnisnya. Hal ini juga terlihat pada perusahaan retail khususnya di Jakarta, perusahaan seperti *Carrefour*, Matahari *Department Store*, *Hypermart*, dan perusahaan sejenis lainnya harus bersaing keras untuk memberikan pelayanan terbaik bagi pelanggannya, agar pelanggan tidak pindah ke pesaing. Salah satu cara untuk memuaskan pelanggan adalah dengan selalu menyediakan stok barang cukup yang dicari oleh pelanggan di tempat retail mereka. Persediaan merupakan bagian dari aktiva lancar perusahaan yang penting, bernilai tinggi serta merupakan harta yang peka terhadap waktu, kerusakan, pencurian, pemborosan, penurunan harga pasar, serta pengeluaran biaya yang berlebihan sehingga biaya pengelolaan persediaan cukup besar. Hal ini dirasakan pada saat suku bunga tinggi, namun sering tidak disadari oleh manajemen. Selain itu juga harus dikeluarkan pula

biaya asuransi, pajak dan biaya penyimpanan dan lain-lain. Perlu juga diperhitungkan kemungkinan adanya kerugian akibat kerusakan, penyusutan, usang dan lain-lain dalam hal pengelolahan persediaan tersebut. Bila stok yang dimiliki perusahaan berkelimpahan, maka perusahaan *retail* tersebut akan merugi, karena barang yang tidak terjual tersebut harus mereka bayar. Sehubungan dengan hal di atas maka dalam makalah ini dibahas optimasi persediaan pada perusahaan retail dengan menggunakan metode *Neuro Dynamic*, yang merupakan penerapan teori *Neural* pada masalah pengendalian persediaan yang dapat menghitung dua atau lebih biaya secara serempak dan juga dapat menentukan stok barang yang optimal.

Untuk membantu proses perhitungan masalah pengendalian persediaan perusahaan retail ini dirancang sebuah program komputer yang dapat menghitung stok barang yang harus disediakan oleh perusahaan retail tersebut agar tidak pernah kehabisan barang (*overdemand*) atau pun barang berkelimpahan (*oversupply*).Program yang telah dibuat dibatasi pada jumlah gudang satu dan jumlah toko tiga buah serta hanya mencari berapa banyak barang yang harus dipesan untuk memenuhi kebutuhan pelanggan, sedangkan *database* untuk barang-barang yang ada di dalam toko, dan faktor-faktor yang dapat menyebabkan distribusi dari satu tempat ke tempat lain tidak lancar, seperti kecelakaan, huru-hara, supir mogok, dan lain sebagainya, dan faktor kenaikan harga belum dapat dihitung.

2. *NEURO-DYNAMIC PROGRAMMING*

Metode *Neuro-Dynamic programming* adalah pengembangan dari *dynamic programming*. Metode ini menggunakan dasar-dasar intelegensi semu (*Artificial Intelligence*) yang mencakup simulasi dan berbasis algoritma serta teknik pendekatan seperti *neural networks*. *Neuro-Dynamic programming* merupakan teknik pendekatan baru dalam bidang pengendalian persediaan (*inventory control*). Metode ini berfokus pada solusi pendekatan yang akan dihasilkan oleh *dynamic programming*, sehingga informasi-informasi yang tidak dipakai didalam d*ynamic programming* dapat dipakai oleh *Neuro-Dynamic programming*. Dalam beberapa tahun terakhir ini, metode *Neuro-Dynamic programming* telah mencatat sejumlah keberhasilan. Misalnya: Contoh kasus yang dilaporkan oleh Bertsekas dan Tsitsiklis (1996) yang mendemonstrasikan keunggulan *Neuro-Dynamic programming*. Walaupun demikian, metode *Neuro-Dynamic programming* masih merupakan metode baru, dan algoritma yang berhasil pada tingkat aplikasi, tidak begitu mudah untuk dimengerti secara keseluruhan pada tingkat teoritis.

Dalam makalah ini, dipakai pendekatan *Neuro-Dynamic programming* untuk meng-optimalkan sistem *inventory retail* (Nahmias dan Smith, 1999). Optimasi sistem *inventory retail* membahas permasalahan pada pemesanan dan penempatan persediaan barang di dalam gudang dan toko untuk memenuhi permintaan pelanggan. Optimasi sistem *inventory retail* secara serempak akan meminimumkan biaya pergudangan dan transportasi. Dalam algoritma *Neuro-Dynamic* untuk tujuan manajemen *inventory retail*, sebenarnya ada dua macam algoritma yaitu algoritma *approximate policy iteration* dan *online temporal difference method.* Namun demikian dalam makalah ini metode yang dipakai adalah *online temporal difference method* karena setiap iterasi, metode ini secara efektif menghitung parameter vektornya, menggunakan *multilayer* untuk menggantikan arsitektur linear dan tingkat eksploratif tinggi.

2.1 *Online Temporal Difference Method*

Algoritma *temporal-difference* sudah diaplikasikan dengan sukses untuk beberapa aplikasi skala besar di dalam *Neuro-Dynamic programming*. Contohnya adalah permainan *Backgammon* (Tesauro, 1998), pengiriman dengan *elevator* (Zhang dan Deitterich, 1996). Adapun proses

Neuro-Dynamic programming yang menggunakan *online temporal difference method* adalah sebagai berikut: (Haykin Simon, 1999).

1. Keadaan sebelum keputusan x_0 adalah sebagai simulator, dan kontrol u_0 dihitung dari
$$u_0 = \min \bar{J}(f_2(x_0, u), r_0) \quad (1)$$
2. Jalankan simulator menggunakan kontrol u_0 untuk mendapatkan keadaan setelah keputusan yang pertama
$$y_0 = f_2(x_0, u_0) \quad (2)$$
3. Secara umum, pada waktu *t*, jalankan simulator menggunakan kontrol u_t untuk mendapatkan keadaan sebelum keputusan selanjutnya
$$x_{t+1} = f_1(y_t, w_t) \quad (3)$$
4. Dapatkan kontrol u_{t+1} dengan cara
$$u_{t+1} = \min \bar{J}(f_2(x_{t+1}, u), r_t) \quad (4)$$
5. Jalankan simulator menggunakan kontrol u_{t+1} untuk mendapatkan keadaan setelah keputusan.
$$y_{t+1} = f_2(x_{t+1}, u_{t+1}) \quad (5)$$
6. Ulangi ke Langkah 3 selama waktu pengiriman yang dibutuhkan.
Dimana *u* = fungsi permintaan ; *x* = fungsi persediaan dan *y* = fungsi biaya.

2.2 Algoritma *Neuro-Dynamic Programming*

Algoritma *Neuro-Dynamic Programing* telah disusun dan dikembangkan berdasarkan rumusan teori yang ada dan setelah disusun diperoleh algoritmanya seperti berikut ini: (Nahmias dan Smith, 1999)

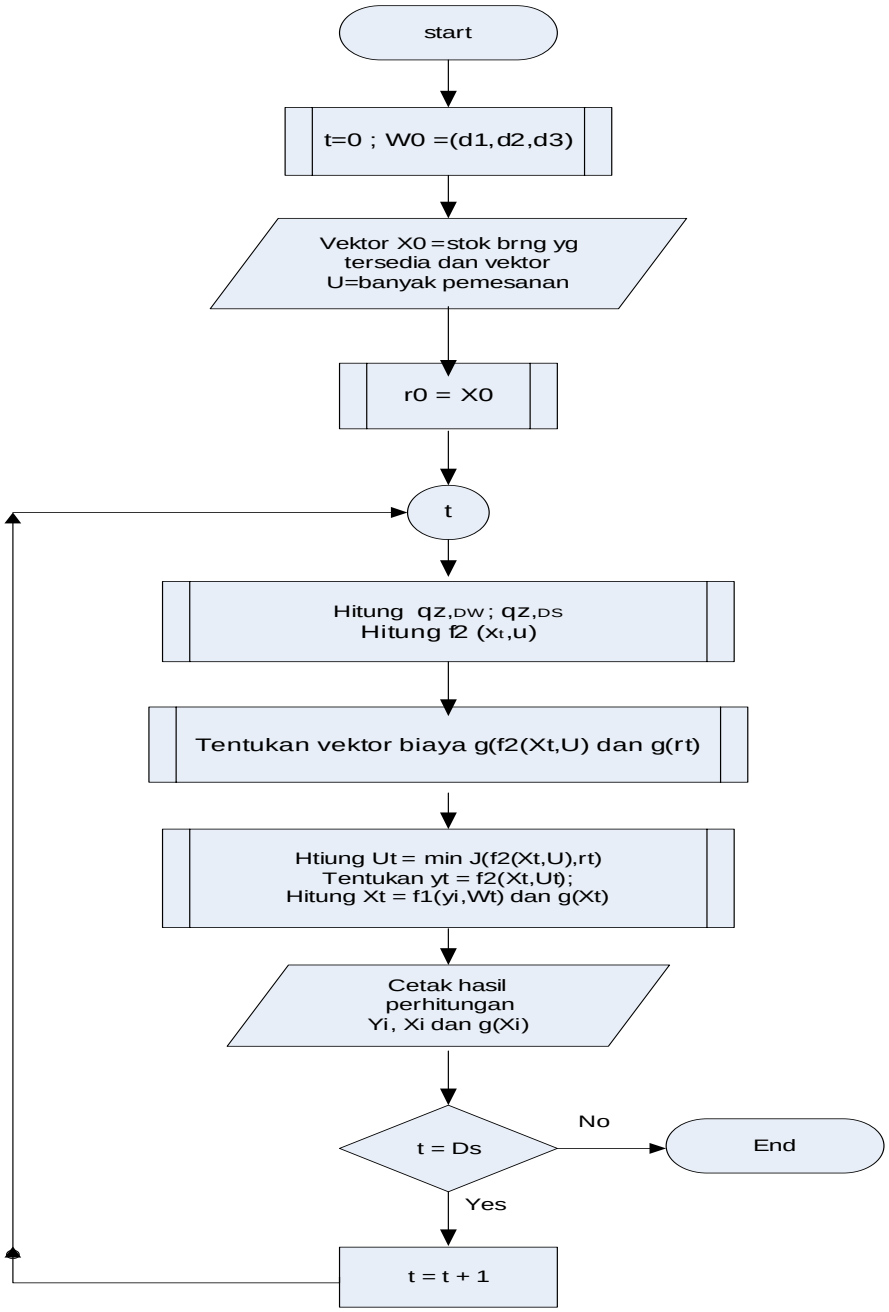
1. Mulai dengan $t = 0$.
2. Tentukan vektor $w_t = (d_1, d_2, d_3)$, besaran d_i merupakan permintaan yang terjadi di toko i . Vektor $w_t = (d_1, d_2, d_3)$ ditentukan dari nilai acak permintaan rata-rata setiap toko.
3. Tentukan vektor $x_t = (q_{0,0}; q_{0,DW}; q_{1,0}; q_{1,DS}; \ldots; q_{K,0}; q_{K,DS})$.
4. Tentukan vektor r_t = vektor $x_t = (q_{0,0}; q_{0,DW}; q_{1,0}; q_{1,DS}; \ldots; q_{K,0}; q_{K,DS})$
5. Tentukan vektor $u = (a_0, a_1, \ldots, a_K)$ berdasarkan banyak pemesanan yang dilakukan oleh bagian pembelian. Besaran a_0 adalah pemesanan untuk gudang dan a_K adalah pemesanan untuk toko *K*.
6. Tentukan vektor $f_2(x_t, u) = (\bar{q}_{0,0}; \bar{q}_{0,Dw}; \bar{q}_{1,0}; \bar{q}_{1,Ds}; \ldots; \bar{q}_{K,0}; \bar{q}_{K,Ds})$.
Dengan $\bar{q}_{0,DW} = q_{0,DW} + a_0; \bar{q}_{0,0} = q_{0,0} - \sum a_i + a_0; \bar{q}_{i,DS} = q_{i,S} + a_{ii}, \forall i \in \{1, \ldots, K\}$, $\bar{q}_{i,0} = q_{i,DS}, \forall i \in \{1, \ldots, K\}$.
7. Tentukan vektor biaya $g(f_2(x_t, u))$
8. Tentukan vektor biaya $g(r_t)$
9. Hitung fungsi $u_t = \min \bar{J}(f_2(x_t, u), r_t)$. Dengan menentukan minimum dari fungsi biaya $J^*(y) = g(f_2(x_0, u))$ dan fungsi biaya $J^*(y) = g(r_t)$.
10. Tentukan vektor $y_t = f_2(x_t, u_t) = (\bar{q}_{0,0}; \bar{q}_{0,Dw}; \bar{q}_{1,0}; \bar{q}_{1,Ds}; \ldots; \bar{q}_{K,0}; \bar{q}_{K,Ds})$ dengan cara yang sama dengan Langkah 6.

11. Tentukan vektor $x_{t+1} = f_{t+1}(y_t, w_t) = \left(\overline{q}_{0,0}; \overline{q}_{0,Dw}; \overline{q}_{1,0}; \overline{q}_{1,Ds}; \ldots; \overline{q}_{K,0}; \overline{q}_{K,Ds}\right)$, dengan $\hat{q}_{i,0} = [\hat{q}_{i,0} - d_i]$, $\forall i \in \{1,\ldots,K\}$, $\hat{q}_{0,0} = 0$, bila permintaan pelanggan dapat dipenuhi oleh persediaan di dalam toko. Bila permintaan pelanggan tidak dapat dipenuhi oleh persediaan di dalam toko maka $\hat{q}_{i,0} = 0$ dan $\hat{q}_{0,0} = \left[\hat{q}_{0,0} + (d_i - C_i)\right]$.
    Lalu $\overline{q}_{0,0} = \overline{q}_{0,0} + q_{0,DW}$, $\overline{q}_{0,Dw} = 0$, $\overline{q}_{i,0} = \hat{q}_{i,0} + q_{i,DS} + (d_i - C_i)$, $\forall i \in \{1,\ldots,K\}$, $\overline{q}_{i,Ds} = 0$.
12. Kemudian dihitung biaya $g(y_t, w_t)$
13. *Update* $t = t + 1$, setelah itu *update* vektor r_t, $f_2(x_t, u)$, $u_t = \min \overline{J}(f_2(x_t, u), r_t)$, $y_t = f_2(x_t, u_t)$, dan $w_t, x_1 = f_t + 1(y_t, w_t)$ dengan cara sama ulangi Langkah 2 sebanyak waktu pengiriman yang dibutuhkan untuk mengirim barang dari gudang ke toko.
14. Algoritma akan berhenti bila jumlah pengulangan proses t = D_S

Keterangan:

x_t	:	variabel keadaan inventory sebelum keputusan	c_s	:	kapasitas toko
y_t	:	variabel keadaan inventory setelah keputusan	u_t	:	variabel keputusan
w_t	:	variabel acak gangguan yang didapat dari distribusi yg ditetapkan, tidak bergantung dari semua informasi yang tersedia samapai kepada waktu *t*	c_p	:	kapasitas produksi (*supplier* menyediakan barang)
c_w	:	kapasitas gudang	d_i	:	banyaknya permintaan pada toko ke-*i* pada suatu hari.
a_0	:	pemesanan untuk gudang	a_k	:	pemesanan untuk toko *k*
J^{uo}	:	fungsi biaya didalam kebijakan			
D_W	:	waktu pengiriman dari pemasok ke gudang	D_S	:	waktu pengiriman dari gudang ke toko *S*
$q_{0,0}$	:	banyak stok barang awal di dalam gudang	$q_{0,DW}$	:	banyak stok barang di dalam gudang pada waktu pengiriman
$q_{K,0}$	:	banyak stok barang awal di dalam toko	K, $q_{K,DW}$	:	banyak stok barang dikurangi dengan permintaan di toko K

Algoritma yang telah diuraikan di atas bila ditulis dalam bentuk *flowchart* program dapat dilihat seperti yang ditampilkan pada Gambar 1.



Gambar 1. *Flowchart* program *Neuro-Dynamic Programming*

3. PERMASALAHAN

Dari analisis sistem yang berjalan di PT CKE TEHNIK permasalahan yang dihadapi adalah:

- Banyak jenis barang yang diinginkan oleh pelanggan tidak terpenuhi oleh barang yang ada di cabang, sehingga memerlukan kiriman khusus dari gudang.

- Terjadi penumpukan barang yang tidak terjual di dalam cabang akibat pemesanan yang terlalu banyak. Pemakaian gudang di kantor cabang tidak efisien dan ongkos untuk penyimpanan barang meningkat.
- Banyak barang yang telah dipesan oleh perusahaan dari pemasok tersimpan terlalu lama dalam gudang karena barang yang dipesan lebih banyak dari pada permintaan konsumen.
- Ketika suatu cabang memesan suatu barang dari gudang, sering terjadi pesanan barang tidak tersedia.
- Banyak pelanggan berpindah ke perusahaan pesaing lain karena barang yang diinginkannya tidak dapat dipenuhi oleh perusahaan. Hal ini disebabkan barang tersebut masih di dalam perjalanan menuju cabang, ataupun karena barang masih ada di gudang.

3.1 Penyelesaian Masalah

Dalam menganalisis bagaimana suatu perusahaan distribusi dapat melakukan pengadaan barang dengan efektif dan efisien, terdapat beberapa hal yang diperlukan. Hal-hal itu adalah menentukan *inventory* yang ada di gudang maupun di toko, model permintaan, jangka waktu pengiriman dan metode-metode yang dapat dilakukan untuk melakukan pemesanan kembali yang efisien. Berikut ini adalah Langkah-Langkah yang dapat dilakukan dalam menganalisis permasalahan tersebut (Roy, 2001).

- Menentukan banyaknya *inventory* yang dibutuhkan.
- Menentukan jumlah pemesanan ekonomis berdasarkan hasil perhitungan program.
- Menentukan frekuensi pemesanan sehingga tidak perlu melakukan pemesanan berulang kali.

Dengan memperhatikan masalah yang di atas, maka dalam makalah ini diusulkan penyelesaiannya dengan metode *Neuro-Dynamic* dan proses perhitungannya dirancang menggunakan program komputer.

3.2 Komponen Rancangan

Komponen utama dalam perancangan program aplikasi *Inventory Control* untuk perusahaan retail dengan metode *Neuro–Dynamic* adalah (Powell, 2003):

a. Modul *input* parameter gudang
b. Modul *input* parameter toko pertama
c. Modul *input* parameter toko kedua
d. Modul *input* parameter toko ketiga
e. Modul *input* pemesanan
f. Modul perhitungan dengan *Neuro-Dynamic*
g. Modul hasil optimasi

3.3 Spesifikasi Rancangan

Perancangan program aplikasi *Inventory Control* untuk perusahaan retail dengan Metode *Neuro–Dynamic* yang akan dirancang dengan spesifikasi sebagai berikut: Bahasa Pemograman: *Borland Delphi* 7. *Platform*: *Microsoft Windows* XP. *Processor*: *Intel Pentium* 4 2.4 GHz. *Memory*: 256 MB. Beberapa rancangan yang telah dibuat untuk program ini antara lain: rancangan layar menu utama, rancangan layar menu *input*an, rancangan layar submenu optimasi dan rancangan layar *output* (Alam, 2003).

4. HASIL DAN PEMBAHASAN

4.1 Hasil

Untuk mendapatkan hasil perhitungan optimasi pengendalian persediaan dengan metode *Neuro-Dynamic programming*, pertama-tama harus diperoleh parameter yang ada di dalam

gudang dengan cara pengamatan, seperti kapasitas gudang, kapasitas pemasok, jumlah stok barang yang tersedia di dalam gudang, biaya penyimpanan per unit, biaya kekurangan barang, biaya pengiriman khusus, dan permintaan rata–rata. Selanjutnya mendapatkan biaya kekurangan barang (*shortage*), besarnya adalah keuntungan yang didapat dari penjualan barang tersebut. Permintaan rata-rata didapatkan dengan cara merata-ratakan permintaan dari pelanggan selama jangka waktu transportasi barang dari gudang ke toko.

Parameter lain yang harus didapatkan adalah banyaknya pemesanan gudang dan pemesanan setiap toko yang biasa dilakukan oleh bagian pembelian. Dalam melakukan pemesanan ada batasan-batasan yang tidak boleh dilampaui. Batasan-batasan itu antara lain: jumlah pemesanan setiap toko tidak dapat melebihi kapasitas toko dikurangi dengan stok barang yang tersedia di dalam toko tersebut. Total pemesanan yang dilakukan oleh toko tersebut tidak bisa melebihi jumlah stok barang yang tersedia di dalam gudang; pemesanan gudang tidak dapat melebihi kapasitas pemasok untuk mengirimkan barang kepada gudang. Sedangkan hasil perhitungan selanjutnya dapat ditunjukan melalui contoh kasus/penerapan berikut ini:

Contoh Kasus/Penerapan

Dari pengamatan setiap hari, kapasitas sebuah gudang diisi dengan suatu jenis kipas angin sebanyak 73 buah, disisi lain pemasok hanya dapat menyediakan 90 buah sedangkan jumlah kipas angin yang tersedia di dalam gudang ada sebanyak 41 buah. Biaya penyimpanan untuk kipas angin jenis tersebut adalah 9000 rupiah per unit per hari. Pengamatan lainnya adalah terhadap tiga toko, yang mana toko pertama terletak di Jalan Hayam Wuruk. Adapun parameter toko tersebut, adalah: kapasitas toko pertama sebanyak 32 buah, jumlah stok barang di dalam toko sebanyak 15 buah, biaya penyimpanan sebesar 15000 rupiah per unit, biaya kekurangan barang (*shortage*) sebesar 5000 rupiah, biaya pengiriman khusus sebesar 30000 rupiah, permintaan rata-rata sebesar 9 buah sampai 29 buah. Kemudian diamati toko kedua yang terletak di Jalan Pluit Raya, didapat parameter toko tersebut, sebagai berikut: kapasitas toko sebanyak 49 buah, jumlah stok barang di dalam toko sebanyak 20 buah, biaya penyimpanan sebesar 17000 rupiah per unit, biaya kekurangan barang (*shortage*) sebesar 60000 rupiah, biaya pengiriman khusus sebesar 30000 rupiah, permintaan rata-rata sebesar 14 buah sampai 45 buah. Selanjutnya pengamatan di toko ketiga yang terletak di Jalan Bandung, didapat parameter toko tersebut sebagai berikut: kapasitas toko sebanyak 15 buah, jumlah stok barang di dalam toko sebanyak 6 buah, biaya penyimpanan sebesar 20000 rupiah per unit, biaya kekurangan barang (*shortage*) sebesar 75000 rupiah, biaya pengirimanan khusus sebesar 40000 rupiah, permintaan rata-rata sebesar 4 buah sampai 12 buah. Berdasarkan pengamatan dari bagian pembelian, nilai pemesanan toko pertama sebesar 14 buah, toko kedua sebesar 19 buah, toko ketiga sebesar 5 buah, dan pemesanan gudang sebesar 38. Lama pengiriman dari gudang ke toko adalah 3 hari. Masalah yang ada dalam contoh ini adalah berapa banyak stok/persediaan yang optimal.

Dari data di atas, jika diproses dengan algoritma *Neuro-Dynamic programming* dapat dilakukan sebagai berikut:

1. Mulai dengan $t = 0$.
2. Tentukan vektor $w_0 = (d_1, d_2, d_3)$. Misalkan setelah vektor w_0 yang diacak didapatkan vektor $w_0 = (10,14,5)$.
3. Tentukan vektor x berdasarkan banyaknya stok barang yang tersedia. Maka vektor $x_0 = (q_{0,0}; q_{0,DW}; q_{1,0}; q_{1,DS}; \ldots; q_{K,0}; q_{K,DS}) = (41,0,15,0,20,0,6,0)$.

4. Tentukan vektor r_0 = vektor
   $x_0 = (q_{0,0}; q_{0,DW}; q_{1,0}; q_{1,DS}; \ldots; q_{K,0}; q_{K,DS}) = (41,0,15,0,20,0,6,0)$.
5. Tentukan vektor $u = (a_0, a_1, \ldots, a_K)$ berdasarkan banyak pemesanan yang dilakukan oleh bagian pembelian. Maka vektor $u = (a_0, a_1, \ldots, a_K) = (38,14,19,5)$.
6. $\bar{q}_{0,DW} = q_{0,DW} + a_0 \rightarrow \bar{q}_{0,3} = 0 + 38 = 38$.

   Jadi $\bar{q}_{0,0} = q_{0,0} - \sum a_i + a_0 = 41 - (14 + 19 + 5) + 38 = 41$

   Untuk setiap toko i, $\bar{q}_{i,DS} = q_{i,S} + a_i \rightarrow \bar{q}_{1,3} = 0 + 14 = 14$, $\bar{q}_{2,3} = 0 + 19 = 19$, $\bar{q}_{3,3} = 0 + 5 = 5$. Terakhir proses mencari $f_2(x_0, u)$ utk setiap toko i, i, $\bar{q}_{i,0} = q_{i,DS} \rightarrow \bar{q}_{1,0} = 15$, $\bar{q}_{2,0} = 20$. $\bar{q}_{3,0} = 6$. Setelah semua tahap tersebut maka ditentukan vektor $f_2(x_0, u)$
   $f_2(x_0, u) = (\bar{q}_{0,0}; \bar{q}_{0,Dw}; \bar{q}_{1,0}; \bar{q}_{1,Ds}; \ldots; \bar{q}_{K,0}; \bar{q}_{K,Ds}) = (41,38,15,14,20,19,6,5)$.
7. Tentukan vektor biaya $g(f_2(x_0, u))$. Karena tidak ada permintaan yang melebihi persediaan barang maka biaya yang terjadi adalah jumlah barang dikali dengan biaya penyimpanan. Maka vektor biaya $g(f_2(x_0, u))$ = (41*9,38*9,15*15,14*15,20*17,19*17,6*20,5*20)
   = (369, 342, 225, 210, 340, 323,120,100).
8. Tentukan vektor biaya $g(r_0)$. Karena tidak ada permintaan yang melebihi persediaan barang maka biaya yang terjadi adalah jumlah barang dikali dengan biaya penyimpanan. Maka vektor biaya
   $g(r_0) = (41*9,0*9,15*15,0*15,20*17,0*17,6*20,0*20) = (369,0,225,0,340,0,120,0)$.
9. Hitung fungsi $u_0 = \min \bar{J}(f_2(x_0, u), r_0)$. Dengan menentukan minimum dari fungsi biaya
   $J^*(y) = g(f_2(x_0, u)) = (369, 342, 225, 210, 340, 323, 120, 100)$ dan fungsi biaya
   $J^*(y) = g(r_0) = (369, 0, 225, 0, 340, 0, 120, 0)$. Kedua nilai tersebut dimasukkan kedalam fungsi $u_0 = \min \bar{J}(f_2(x_0, u), r_0) = \min \bar{J}(369,369;342,0;225,225;210,0;340,340;323,0;120,120;100,0) = (369,0;225,0;340,0;120,0) = (41,0;15,0;20,0;6,0) = (41,15,20,6)$.
10. Tentukan vektor $y_0 = f_2(x_0, u_0) = (\bar{q}_{0,0}; \bar{q}_{0,Dw}; \bar{q}_{1,0}; \bar{q}_{1,Ds}; \ldots; \bar{q}_{K,0}; \bar{q}_{K,Ds})$. Dengan cara yang sama dengan Langkah 6 maka prosesnya adalah sebagai berikut:
    $\bar{q}_{0,DW} = q_{0,DW} + a_0 \rightarrow \bar{q}_{0,3} = 0 + 41 = 41$.
    Jadi $\bar{q}_{0,0} = q_{0,0} - \sum a_i + a_0 = 41 - (15 + 20 + 6) + 41 = 41$. Lalu untuk setiap toko i, i, $\bar{q}_{i,DS} = q_{i,DS} + a_i \rightarrow \bar{q}_{1,3} = 0 + 15 = 15, \bar{q}_{2,3} = 0 + 20 = 20, \bar{q}_{3,3} = 0 + 6 = 6$. Terakhir adalah proses mencari $y_0 = f_2(x_0, u_0)$ untuk setiap toko i, $\bar{q}_{i,0} = q_{i,0} \rightarrow \bar{q}_{1,0} = 15$, $\bar{q}_{2,0} = 20$, $\bar{q}_{3,0} = 6$. Setelah semua tahap tersebut maka fungsi
    $y_0 = f_2(x_0, u_0) = (\bar{q}_{0,0}; \bar{q}_{0,Dw}; \bar{q}_{1,0}; \bar{q}_{1,Ds}; \ldots; \bar{q}_{K,0}; \bar{q}_{K,Ds}) = (41, 41, 15, 15, 20, 20, 6, 6)$.
11. Setelah itu dicari vektor $x_1 = f_1(y_0, w_0)$. Jika tidak ada permintaan yang melebihi stok barang, maka untuk setiap toko i dan semua permintaan dapat dipenuhi oleh stok di dalam toko, sehingga nilai $\hat{q}_{0,0} = 0$. Nilai $\hat{q}_{i,0} = [q_{i,0} - d_i] \rightarrow \hat{q}_{1,0} = \lfloor q_{1,0} - d_1 \rfloor = 15 - 10 = 5$, $\hat{q}_{2,0} = \lfloor q_{2,0} - d_2 \rfloor = 20 - 14 = 6$, $\hat{q}_{3,0} = \lfloor q_{3,0} - d_3 \rfloor = 6 - 5 = 1$. Kemudian dihitung untuk setiap

toko $i, \overline{q}_{i,0} = \hat{q}_{i,0} + q_{i,3} \rightarrow \overline{q}_{1,0} = \hat{q}_{1,0} + q_{1,3} = 5 + 15 = 20, \overline{q}_{2,0} = \hat{q}_{2,0} + q_{2,3} = 6 + 20 = 26,$ $\overline{q}_{3,0} = \hat{q}_{3,0} + q_{3,3} = 6 + 1 = 7$. Lalu ditetapkan $\overline{q}_{1,3} = 0$, $\overline{q}_{2,3} = 0$, dan $\overline{q}_{3,3} = 0$. Setelah itu dicari $\overline{q}_{0,0} = \hat{q}_{0,0} + q_{0,3} = 0 + 41 = 41$. Lalu ditetapkan $\overline{q}_{0,3} = 0$, sehingga vektor $x_1 = f_1(y_0, w_0) = (41,0,20,0,26,0,7,0)$.

12. Dari vektor x_1 dihitung ongkos yang terjadi yaitu
$g(x_1) = (41*9,0*9,20*15,0*15,26*17,0*17,7*20,0*20) = (369,0,300,0,442,0,140,0)$
dengan total biaya yang harus dikeluarkan sama dengan 369 + 0 + 300 + 0 + 442 + 0 + 140 + 0 = 1251 ribu rupiah; banyak stok barang didalam gudang sebanyak 41 buah, toko pertama sebanyak 20, toko kedua sebanyak 26, toko ketiga sebanyak 7.
13. Setelah diperoleh vektor x_1 maka selesailah proses *training* pertama. Setelah itu diperlukan *training* sebanyak waktu pengiriman dalam hal ini 3 kali. Maka sekarang dimulai proses *training* ke-2 dengan $t = 1$. Vektor w_1 yang diacak didapatkan vektor $w_1 = (17,20,4)$. Vektor r_1 = vektor $x_1 = (41,0,20,0,26,0,7,0)$. Dipakai $u = (a_0, a_1, \ldots, a_K) = (38,14,19,5)$. Kemudian dari vektor x_1 dan u dicari $f_2(x_1, u) = (41,38,20,14,26,19,7,5)$. Kemudian cari fungsi biaya $g(f_2(x_1, u))$ dan fungsi biaya $g(r_1)$ kemudian kedua nilai tersebut dimasukkan ke dalam fungsi $u_1 = \min \overline{J}(f_2(x_1, u_0), r_1)$. Setelah itu didapatkan $u_1 = (41,20,26,7)$. Langkah selanjutnya adalah menentukan fungsi $y_1 = f_2(x_1, u_1)$. Dengan cara yang sama dengan mencari fungsi $f_2(x_1, u)$, maka fungsi $y_1 = f_2(x_1, u_1) = (41,41,20,20,26,26,7,7)$. Setelah itu dicari $x_2 = f_1(y_1, w_1) = (41,0,23,0,32,0,10,0)$. Dari vektor x_2 dihitung ongkos yang terjadi yaitu $g(x_2) = (41*9,0*9,23*15,0*15,32*17,0*17,10*20,0*20) = (369,0,345,0,544,0,200,0)$
dengan total biaya yang harus dikeluarkan sama dengan 369+0+345+0+544+0+200+0 = 1458 ribu rupiah, banyak stok barang didalam gudang sebanyak 41 buah, toko pertama sebanyak 23, toko kedua sebanyak 32, toko ketiga sebanyak 10.
14. Setelah mendapatkan vektor x_2 maka selesai proses *training* kedua. Sekarang dimulai proses *training* yang terakhir dalam soal ini dengan $t = 2$. Vektor w_2 yang diacak didapatkan vektor w_2 = (19,23,9). Vektor r_2 = vektor x_2 = (41,0,23,0,32,0,10,0). Dipakai $u = (a_0, a_1, \ldots, a_K)$ = (38,14,19,5). Kemudian dari vektor x_2 dan u dicari $f_2(x_2, u) = (41,38,23,14,32,19,10,5)$. Kemudian cari fungsi biaya $g(f_2(x_2, u))$ dan fungsi biaya $g(r_2)$ kemudian kedua nilai tersebut dimasukkan ke dalam fungsi $u_2 = \min \overline{J}(f_2(x_2, u_1), r_2)$. Setelah itu didapatkan $u_2 = (41,23,32,10)$. Langkah selanjutnya adalah menentukan fungsi $y_2 = f_2(x_2, u_2)$. Dengan cara yang sama dengan mencari fungsi $f_2(x_2, u)$, maka fungsi $y_2 = f_2(x_2, u_2)$= (41,41,23, 23,32,32,10,10). Setelah itu dicari $x_3 = f_1(y_2, w_2) = (41,0,27,0,41,0,11,0)$.
15. Dari vektor x_2 dihitung ongkos yang terjadi yaitu $g(x_2)$= (41*9,0*9,27*15,0*15,41*17, 0*17,11*20,0*20) = (369,0,405,0,697,0,220,0) dengan total biaya yang harus dikeluarkan sama dengan 69+0+405+0+697+0+220+0 = 1691 ribu rupiah, banyak stok barang didalam gudang sebanyak 41 buah, toko pertama sebanyak 27, toko kedua sebanyak 41, toko ketiga sebanyak 11.

Bila permasalahan di atas diselesaikan dengan komputer diperoleh beberapa tampilan *output* seperti berikut:

Gambar 4. Tampilan halaman parameter gudang

Gambar 5. Tampilan halaman parameter Toko 1

Gambar 6. Tampilan halaman optimasi

4.2 Pembahasan

Berikut adalah tabel laporan mingguan untuk kipas jenis DFT-2.5# PT CKE TEHNIK setelah memakai metode *Neuro-Dynamic programing* dan sebelum memakai metode ini.

Tabel 1. Laporan mingguan DFT-2.5# sebelum memakai metode

	Keterangan	Jumlah
Toko 1	Penjualan	153 buah
	Stok barang yang tersisa	23 buah
	Permintaan yang tidak dapat terpenuhi	12 buah
Toko 2	Penjualan	201 buah
	Stok barang yang tersisa	29 buah
	Permintaan yang tidak dapat terpenuhi	24 buah
Toko 3	Penjualan	103 buah
	Stok barang yang tersisa	6 buah
	Permintaan yang tidak dapat terpenuhi	5 buah

Tabel 2. Laporan mingguan DFT-2.5# setelah memakai metode

	Keterangan	Jumlah
Toko 1	Penjualan	163 buah
	Stok barang yang tersisa	27 buah
	Permintaan yang tidak dapat terpenuhi	5 buah
Toko 2	Penjualan	221 buah
	Stok barang yang tersisa	30 buah
	Permintaan yang tidak dapat terpenuhi	4 buah
Toko 3	Penjualan	142 buah
	Stok barang yang tersisa	10 buah
	Permintaan yang tidak dapat terpenuhi	1 buah

Pembahasan dari dua tabel diatas menunjukkan bahwa dengan pamakaian metode *Neuro-Dynamic programing* diperoleh beberapa informasi antara lain: penjualan meningkat setelah pemakaian metode di atas karena jumlah permintaan yang tidak dapat dipenuhi dapat diminimalkan, sehingga order pemesanan bertambah mengakibatkan keuntungan menjadi bertambah pula.

- Stok barang yang disediakan baik di toko maupun di gudang cukup.
- Jumlah permintaan pelanggan atas suatu jenis kipas yang tidak terpenuhi dapat diminimalkan.
- Secara keseluruhan keuntungan yang didapat optimal/meningkat.

5. KESIMPULAN

Berdasarkan pembahasan yang telah dilakukan terhadap sistem yang sedang berjalan pada PT CKE TEHNIK, maka dapat ditarik kesimpulan sebagai berikut: pemesanan barang berdasarkan metode *Neuro-Dynamic* cukup membantu dalam pengoperasian sistem bisnis PT CKE TEHNIK. Metode ini telah dapat membantu para staf pembelian dalam mengambil keputusan taktis karena tersedianya informasi yang dibutuhkan secara cepat dan cukup akurat. Penentuan stok barang berdasarkan metode *Neuro-Dynamic* dapat mengoptimalkan jumlah stok

barang yang ada, sehingga tidak terjadi *overdemand* dan *oversupply*. Selanjutnya untuk pengembangannya disarankan digunakan pada jenis *computer system mobile,* agar lebih fleksibel bisa digunakan baik di lapangan maupun bukan.

Ucapan Terima Kasih:

Pada kesempatan ini penulis mengucapkan banyak terima kasih kepada Hiwanto**,** yang telah membantu dalam pembuatan program makalah ini, sehingga makalah ini dapat terselesaikan dengan baik.

DAFTAR PUSTAKA

Alam, M.A.J., 2003. *Membuat Program Aplikasi Menggunakan Delphi 6 & 7*, Gramedia, Jakarta.

Bertsekas, D.P., 2005. *A Neural-Dynamic Programming Approach to Retailer Management*, USA.

Hansen, E., 2004. *Dynamic Programming for Partially Observable Stochastic Games*, Missisippi.

Haykin, S., 1999. *Neural Networks*, Prentice Hall Inc., New Jersey.

Nahmias and Smith, 1999, *A Neuro Dynamic Programming Approach to Retailer Inventory Management*,. http://citeseer.ist.psu.edu/.

Powell, W., 2003. *Approximate Dynamic Programming for Asset Management*, Princeton.

Roy, V., 2001. *Inventory Management*, Unica Technologies Inc., Lincoln, USA.

Tsitsiklis J., 1996. *Neuro Dynamic Programming*, Prentice Hall Inc., USA.

Tesauro, 1998. "Co-Evolution in the Successful Learning of Backgammon Strategy." *Machine Learning*, Vol. 32, September.

Zhang and Deitterich, 1996. *A Reinforcement Learning Approach to Job Shop Scheduling*, http://citeseer.ist.psu.edu/.