Løsningsforslag til Øving 7:

Oppgave 1:

a) **Hastighetsøkning :** Forholdet mellom den tiden det tar å utføre en oppgave sekvensielt på en uniprosessor og den tiden det tar å utføre den parallelt på et n-prosessor-system, altså et mål på hvor mye raskere det går å utføre oppgaven parallelt.

Effektivitet : Forholdet mellom det antallet operasjoner en uniprosessor må utføre for å gjennomføre en beregning og det antallet enhetsoperasjoner en n-prosessor maksimalt kan utføre i løpet av den tiden $T(n)$ den bruker på den samme beregningen, og er derfor et mål på hvor effektivt regnekapasiteten i n-prosessoren utnyttes ved den parallelle beregningen.

Redundans : Måler hvor mye beregningsbehovet for å løse problemet øker ved å utføre beregningen i parallell.

Utnyttelse : Forventet andel av tiden hver av de n prosessorene er opptatt med beregninger.

Kvalitet : Produktet av hastighetsøkning, effektivitet og den inverse av redundansen. Det er ønskelig at de to første er så høye som mulig, mens redundansen er så lav som mulig. Kvaliteten er høy når dette er oppfylt, og er et slags samlemål på godheten av den parallelle beregningen.

b) (a) $S(n)$ antar sitt minimum når beregningen overhodet ikke inneholder noe parallellitet, dvs. når operasjonene må utføres sekvensielt Da er tidsforbruket på en uniprosessor lik tidsforbruket på en n-prosessor: $T(n) = T(1) \Rightarrow S(n) = 1$.

S(n) antar sitt maksimum når beregningen kan deles opp i n helt uavhengige delberegninger, som hver kan kjøres på en av de n prosessorene. Da er $T(n) = \frac{1}{n} \cdot T(1) \Rightarrow S(n) = n$. Altså er :

$$\underline{1 \leq S(n) \leq n}$$

(b)

$$E(n) = \frac{1}{n} \cdot \frac{T(1)}{T(n)} = \frac{1}{n} \cdot S(n)$$

$$\Downarrow$$

$$\underline{\frac{1}{n} \leq E(n) \leq 1}$$

(c)

$$R(n) = \frac{P(n)}{P(1)} \quad , \quad U(n) = \frac{P(n)}{n \cdot T(n)}$$

$$\Downarrow$$

$$U(n) = \frac{P(n)}{n \cdot T(n)} = \frac{P(n)}{P(1)} \cdot \frac{P(1)}{n \cdot T(n)} = R(n) \cdot \frac{P(1)}{n \cdot T(n)}$$

$$\Downarrow_{(P(1)=T(1))}$$

$$\underline{U(n)} = R(n) \cdot \frac{T(1)}{n \cdot T(n)} = \underline{R(n) \cdot E(n)}$$

(d)

$$\underline{Q(n)} = \frac{T(1)^3}{n \cdot T(n)^2 \cdot P(n)} = \frac{T(1)}{T(n)} \cdot \frac{T(1)}{n \cdot T(n)} \cdot \frac{T(1)}{P(n)} = \underline{\frac{S(n) \cdot E(n)}{R(n)}}$$

(e)

$$1 \leq S(n) \leq n$$

$$\Downarrow$$

$$1 \leq \frac{T(1)}{T(n)} \leq n$$

$$\Downarrow$$

$$\frac{1}{n} \leq \frac{T(1)}{n \cdot T(n)} \leq 1$$

$$\Downarrow$$

$$\frac{1}{n} \leq E(n) \leq 1$$

Totalt antall enhetsoperasjoner kan ikke være mindre ved den parallelle beregningen enn den sekvensielle : $P(n) \geq P(1)$. Dette medfører at

$$U(n) = \frac{P(n)}{n \cdot T(n)} \geq \frac{P(1)}{n \cdot T(n)} = E(n)$$

Den parallelle beregningen medfører totalt $P(n)$ enhetsoperasjoner, som hver tar én tidsenhet. Dersom disse operasjonene var helt uavhengige, ville de kunne utføres i løpet av $P(n)/n$ tidsenheter på de n prosessorene. Dersom det er avhengigheter tilstede vil tidsforbruket ($T(n)$) ta lengre tid. Altså :

$$T(n) \geq \frac{P(n)}{n}$$

$$\Downarrow$$

$$U(n) = \frac{P(n)}{n \cdot T(n)} \leq 1$$

Disse tre resultatene viser at :

$$\underline{\frac{1}{n} \leq E(n) \leq U(n) \leq 1}$$

(f)

$$P(n) \geq P(1) \Rightarrow R(n) = \frac{P(n)}{P(1)} \geq 1$$

$$E(n) = \frac{T(1)}{n \cdot T(n)} \leq \frac{T(1)}{P(n)} = \frac{P(1)}{P(n)} = \frac{1}{R(n)}$$

$$\Downarrow$$

$$R(n) \leq \frac{1}{E(n)}$$

$$\frac{1}{n} \leq E(n) \leq 1 \Rightarrow 1 \leq \frac{1}{E(n)} \leq n$$

Disse tre relasjonene kan sammenfattes i :

$$\underline{1 \leq R(n) \leq \frac{1}{E(n)} \leq n}$$

(g)

$$Q(n) = \frac{T(1)}{T(n)} \cdot \frac{T(1)}{n \cdot T(n)} \cdot \frac{T(1)}{P(n)} = \frac{T(1)}{T(n)} \cdot \frac{T(1)}{n \cdot T(n)} \cdot \frac{P(1)}{P(n)}$$

$$\Updownarrow$$

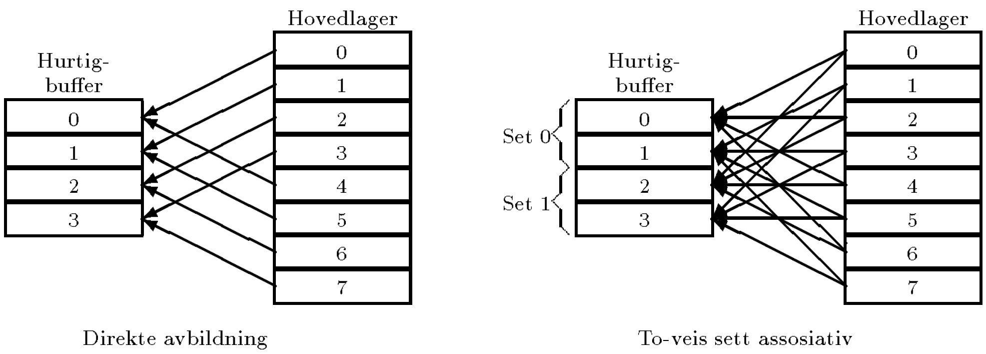
$$Q(n) = S(n) \cdot E(n) \cdot \frac{1}{R(n)}$$

Siden $E(n) \leq 1$, $1/R(n) \leq 1$ og $S(n) \leq n$ må vi ha at :

$$\underline{Q(n) \leq S(n) \leq n}$$

Oppgave 2:

a) Avbildningene er vist i figuren under:



b) Hurtigbuffer-koherens, direkteavbildet:

- Blank: ingen endring eller tom blokk.
- RO n: Read only, block #n
- RW n: Read/write, block #n
- INV n: Invalidate block #n
- BF n: Fetch block #n
- WB n: Write back block #n
- Arb : Arbitrate

	Hurtigbuffer 1					Hurtigbuffer 2					Buss-
Sykel	0	1	2	3	Hva?	0	1	2	3	Hva?	tilstand
1					Arb					Arb	Contention
2	RO 0				BF 0					Wait	Busy
3	CPU 0							RO 2		BF 2	Busy
4	RW 0				Hit			CPU 2			Idle
5	RW 0				Hit			RW 2		Hit	Idle
6					Arb					Arb	Contention
7		RW 1			BF 1					Wait	Busy
8		CPU 1								Wait	Idle
9	INV 0				WB 0						Busy
10		RW 1			Hit	RO 0				BF 0	Busy
11	RO 4				BF 4	CPU 0					Busy
12	CPU 4					RO 0				Hit	Idle
13					Arb					Arb	Contention
14				RO 3	BF 3					Wait	Busy
15				CPU 3					RW 7	BF 7	Busy
16				RO 3	Hit				CPU 7		Hit
17					Arb					Arb	Contention
18					WB 1					Wait	Busy
19		RO 5			BF 5					Wait	Busy
20		CPU 5					RO 5			BF 5	Busy
21		RW 5			Hit		CPU 5				Idle
22		RW 5			Hit		INV 5				Busy
23					WB 5						Busy
24							RO 5			BF 5	Busy
25							CPU 5				Idle
26							RO 5			Hit	Idle
27									RW 7	Hit	Idle
28									RW 7	Hit	Idle
29						RO 0				Hit	Idle

c) Hurtigbuffer-koherens, 2-veis sett-assosiativ, LRU erstatnings-strategi:

Sykel	Hurtigbuffer 1					Hurtigbuffer 2					Buss-
	0	1	2	3	Hva?	0	1	2	3	Hva?	tilstand
1					Arb					Arb	Contention
2	RO 0				BF 0					Wait	Busy
3	CPU 0							RO 2		BF 2	Busy
4	RW 0				Hit			CPU 2			Idle
5	RW 0				Hit			RW 2		Hit	Idle
6					Arb					Arb	Contention
7		RW 1			BF 1					Wait	Busy
8		CPU 1								Wait	Idle
9	INV 0				WB 0						Busy
10		RW 1			Hit	RO 0				BF 0	Busy
11			RO 4		BF 4	CPU 0					Busy
12			CPU 4			RO 0				Hit	Idle
13					Arb					Arb	Contention
14				RO 3	BF 3					Wait	Busy
15				CPU 3					RW 7	BF 7	Busy
16				RO 3	Hit				CPU 7		Hit
17					Arb					Arb	Contention
18		RO 5			BF 5					Wait	Busy
19		CPU 5					RO 5			BF 5	Busy
20		RW 5			Hit		CPU 5				Idle
21		RW 5			Hit		INV 5				Busy
22					WB 5						Busy
23							RO 5			BF 5	Busy
24							CPU 5				Idle
25							RO 5			Hit	Idle
26									RW 7	Hit	Idle
27									RW 7	Hit	Idle
28						RO 0				Hit	Idle