UNIVERSIDAD DE GRANADA

Simulación de Sistemas

Ejercicio de Modelos Discretos

Alejandro Manzanares Lemus

alexmnzlms@correo.ugr.es

1 de febrero de 2021

Índice general

Ejercicio 1:
Modelos discretos

1.1: Simulador base

El código del simulador de esta compañía esta disponible en **src/compania.cpp** e **include/compania.h**

1.1.1: Variables de interés

- **tam**: Representa el tamaño de la demanda del producto **D**.
- **nivel**: Representa el nivel de inventario **I(t)**.
- **pedido**: Representa el volumen de producto que se va a pedir **Z**.
- **tultsuc**: Representa el tiempo ente el suceso actual y el suceso anterior.
- **spequena**: Representa **s**.
- **sgrande**: Representa **S**.

1.1.2: Lista de sucesos

La lista de sucesos se representa de la siguiente manera:

```
typedef struct {
        int suceso;
        float tiempo;
        registro reg_cola;
} suc;

suc nodo;

std::list<suc> lsuc;
```

Simplemente se trata de una lista de objetos del tipo suc, que almacenan el tipo de suceso y el tiempo en el que está planificado. Existe un suceso nodo, que se utiliza para almacenar el suceso actual en cada momento.

1.1.3: Grafo de sucesos

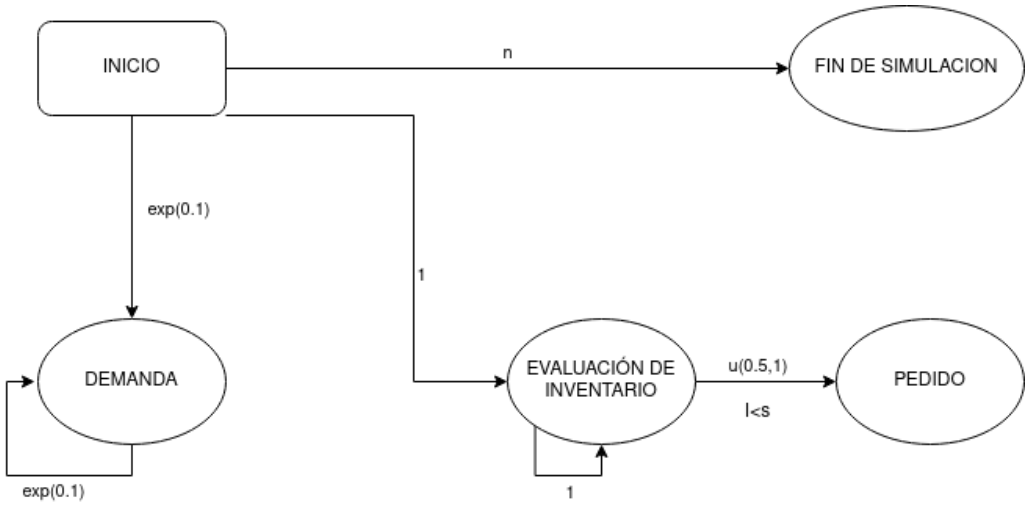

Figura 1.1: Grafo de sucesos

1.1.4: Sucesos

A continuación se muestran los sucesos que se han considerado:

Suceso demanda

Simula la llegada de una demanda del producto a la empresa.

```
if(nivel > 0){
        acummas += (reloj-tultsuc)*nivel;
} else {
        acummenos += (reloj-tultsuc)*(-nivel);
}
tultsuc = reloj;
tam = genera_tamano();
nivel -= tam;
nodo.suceso = SUCESO_DEMANDA;
nodo.tiempo = reloj+gendem(0.1);
insertar_lsuc(nodo);
```

El nivel de inventario disminuye tanto como tamaño tenga la demanda del producto.

Suceso evaluación de inventario

Simula la revisión mensual del inventario por parte de la empresa.

```
if(nivel < spequena && pedido == 0){
        pedido = sgrande - nivel;
        acumpedido += K+i*pedido;
        nodo.suceso = SUCESO_PEDIDO;
        nodo.tiempo = reloj+genpedido(0.5, 1.0);
        insertar_lsuc(nodo);
}
nodo.suceso = SUCESO_EVAL;
nodo.tiempo = reloj+1.0;
insertar_lsuc(nodo);
```

Si $I < s$ y no hay ninguna cantidad de pedido, entonces se hace un pedido de tamaño $S - I$.

Suceso pedido

Simula la llegada de un pedido encargado por la empresa.

```
if(nivel > 0){
        acummas += (reloj-tultsuc)*nivel;
} else {
        acummenos += (reloj-tultsuc)*(-nivel);
}
tultsuc = reloj;
nivel += pedido;
pedido = 0;
```

El nivel de inventario aumenta según el tamaño del pedido.

1.1.5: Generador de informes

Las contadores estadísticos que se utilizan para la generación de informes son:

- **acummas**: Contador estadístico que representa el coste de mantenimiento $I(t)^+$.
- **acummenos**: Contador estadístico que representa el coste de déficit $I(t)^-$.
- **acumpedido**: Contador estadístico que representa el coste de pedido **K+iZ**.

Los informes se almacenan en una matriz de datos, en los que cada fila es un linea del tipo:

```
{(s,S), acumpedido + acummas + acummenos, acumpedido, acummas, acummenos};
```

Además finalmente se muestra la combinación de (s,S) que obtiene un valor de coste total más pequeño.

Los valores aportados son la media de todas las simulaciones realizadas.

1.1.6: Resultados de la simulación

Simulaciones realizadas: 100000

Política	Costo Total	Costo de pedido	Costo de mantenimiento	Costo de déficit
(0,40)	104.232	87.9343	5.80992	10.488
(0,60)	105.026	84.3331	12.9841	7.70838
(0,80)	109.766	82.4005	21.2844	6.08128
(0,100)	116.356	81.2058	30.1149	5.03571
(20,40)	110.368	97.6004	9.12237	3.64483
(20,60)	108.607	88.5082	17.5176	2.58156
(20,80)	113.672	84.9072	26.8582	1.90621
(20,100)	120.901	82.9457	36.4555	1.4997
(40,60)	124.186	98.2922	25.5252	0.368201
(40,80)	124.355	89.1253	34.9559	0.273466
(40,100)	130.665	85.4697	44.9903	0.204556
(60,80)	143.768	98.8523	44.9009	0.0144457
(60,100)	144.116	89.6499	54.455	0.0107663

El mínimo es 104.232 y se obtiene para la configuración (0,40).

1.2: Primera modificación

1.2.1: Variables de interés

- **vendido_mes**: Representa la cantidad de producto vendida en el mes anterior al actual.

1.2.2: Grafo de sucesos

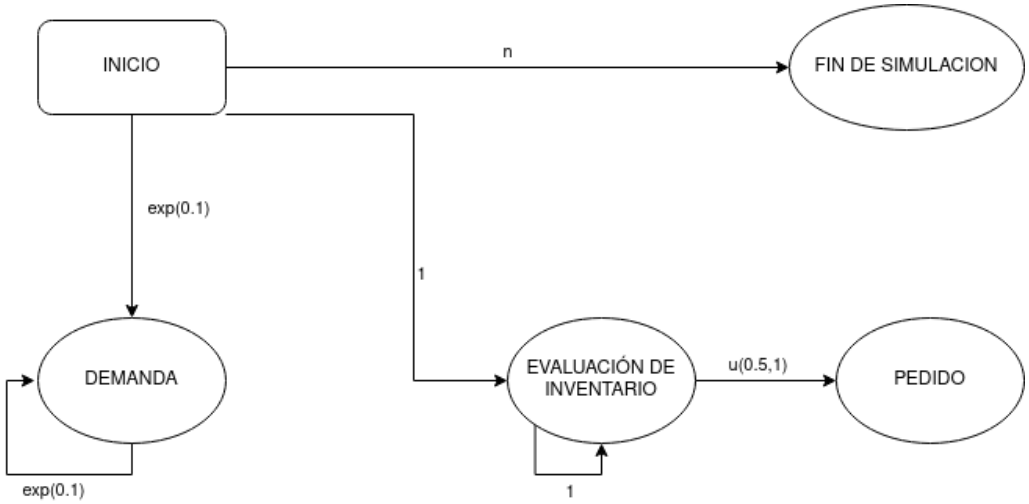

Figura 1.2: Grafo de sucesos

1.2.3: Sucesos

Suceso demanda

```
if(nivel > 0){
        acummas += (reloj-tultsuc)*nivel;
} else {
        acummenos += (reloj-tultsuc)*(-nivel);
}
tultsuc = reloj;
tam = genera_tamano();
nivel -= tam;
vendido_mes += tam;
nodo.suceso = SUCESO_DEMANDA;
nodo.tiempo = reloj+gendem(0.1);
insertar_lsuc(nodo);
```

Se guarda la cantidad de producto vendido durante el mes actual.

Suceso evaluación de inventario

```
pedido = vendido_mes;
vendido_mes = 0;
acumpedido += K+i*pedido;
nodo.suceso = SUCESO_PEDIDO;
nodo.tiempo = reloj+genpedido(0.5, 1.0);
insertar_lsuc(nodo);
nodo.suceso = SUCESO_EVAL;
nodo.tiempo = reloj+1.0;
insertar_lsuc(nodo);
```

El pedido del mes es la cantidad de producto vendida el mes anterior.

Suceso pedido

No ha sido necesario realizar cambios en este suceso.

1.2.4: Resultados de la simulación

Simulaciones realizadas: 100000

Política	Costo Total	Costo de pedido	Costo de mantenimiento	Costo de déficit
(0,40)	135.105	105.974	29.0322	0.0990368
(0,60)	135.114	105.988	29.027	0.0989829
(0,80)	135.121	106.002	29.0191	0.0993632
(0,100)	135.118	105.994	29.0242	0.0991591
(20,40)	135.115	105.992	29.0239	0.0992819
(20,60)	135.115	105.991	29.0245	0.0993882
(20,80)	135.107	105.976	29.0317	0.0988683
(20,100)	135.122	106.003	29.0191	0.0994772
(40,60)	135.116	105.987	29.0295	0.0989866
(40,80)	135.12	106.001	29.0199	0.0998262
(40,100)	135.112	105.986	29.0272	0.0990688
(60,80)	135.115	105.99	29.0259	0.0993546
(60,100)	135.116	105.992	29.025	0.0990352

El mínimo es 135.105 y se alcanza en la configuración (0,40).

1.3: Segunda modificación

1.3.1: Variables de interés

- **pedido_encargado**: Representa si hay encargado un pedido o no.

1.3.2: Grafo de sucesos

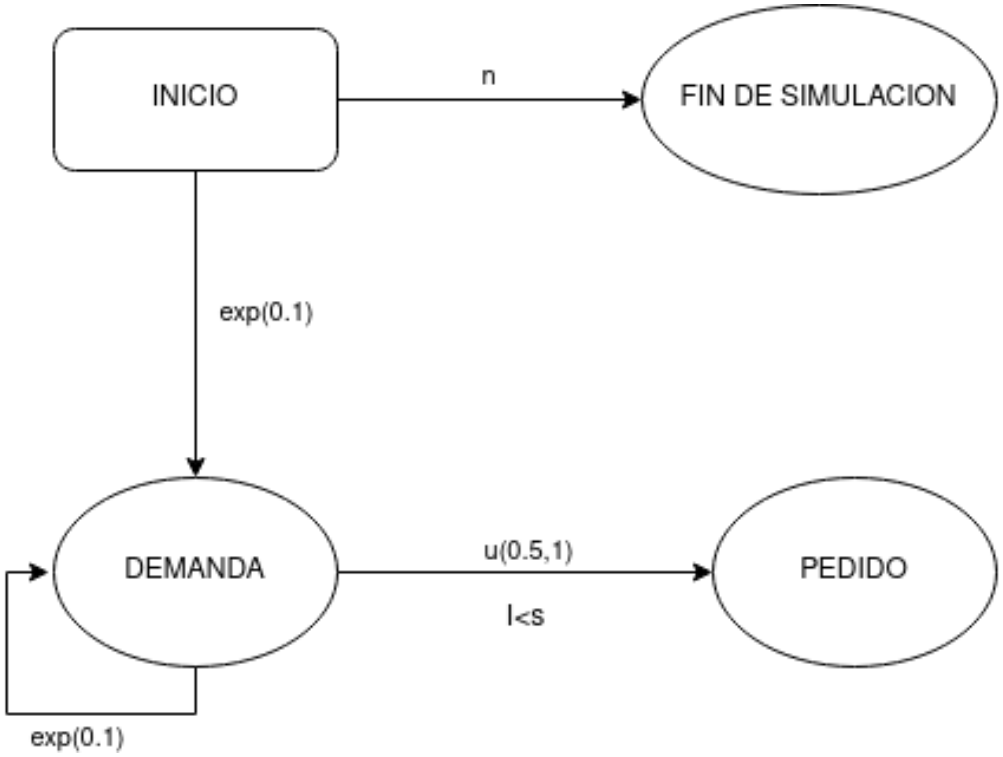

Figura 1.3: Grafo de sucesos

1.3.3: Sucesos

Suceso demanda

```
if(nivel > 0){
        acummas += (reloj-tultsuc)*nivel;
} else {
        acummenos += (reloj-tultsuc)*(-nivel);
}
tultsuc = reloj;
tam = genera_tamano();
nivel -= tam;
if(nivel < spequena && !pedido_encargado){
        pedido = sgrande - nivel;
        acumpedido += K+i*pedido;
        nodo.suceso = SUCESO_PEDIDO;
        nodo.tiempo = reloj+genpedido(0.5, 1.0);
        insertar_lsuc(nodo);
        pedido_encargado = true;
}
nodo.suceso = SUCESO_DEMANDA;
```

```
nodo.tiempo = reloj+gendem(0.1);
insertar_lsuc(nodo);
```

Siempre que no haya un pedido encargado e $I < s$, se hace un pedido de tamaño $S - I$.

Suceso pedido

```
if(nivel > 0){
        acummas += (reloj-tultsuc)*nivel;
} else {
        acummenos += (reloj-tultsuc)*(-nivel);
}
tultsuc = reloj;
nivel += pedido;
pedido = 0;
pedido_encargado = false
```

Una vez llega el pedido, se puede volver a realizar otro.

1.3.4: Resultados de la simulación

Simulaciones realizadas: 100000

Política	Costo Total	Costo de pedido	Costo de mantenimiento	Costo de déficit
(0,40)	105.342	92.8309	7.37193	5.13969
(0,60)	106.048	87.0459	15.5264	3.47599
(0,80)	111.337	84.2107	24.4896	2.63657
(0,100)	118.516	82.5862	33.8031	2.12676
(20,40)	119.046	106.049	11.5772	1.41913
(20,60)	116.147	93.4654	22.2841	0.39756
(20,80)	119.939	87.6403	32.0333	0.265872
(20,100)	126.819	84.7803	41.8365	0.201937
(40,60)	136.747	106.777	29.9181	0.0524362
(40,80)	135.767	94.0802	41.6825	0.0045564
(40,100)	139.847	88.2189	51.6256	0.00291496
(60,80)	157.182	107.478	49.7034	0.000691493
(60,100)	156.292	94.699	61.5931	5.76485e-06

El mínimo es 105.342 y se alcanza en la configuración (0,40).

1.4: Conclusión

Como se puede observar en los apartados correspondientes a los resultados de las simulaciones, todas ellas alcanzan el coste mínimo para la configuración (0,40). Esto significa:

- En la versión base, cada vez que se programa una evaluación, si el nivel del inventario es negativo, se hace un pedido de 40 más el déficit del inventario en el momento de la revisión.
- No afecta a la primera modificación, puesto que depende de la demanda.
- En la segunda modificación, implica que cada vez que el nivel del inventario es negativo (menor que 0), se hace un pedido de 40 más el déficit del producto, con esto se consigue que siempre haya como mínimo un stock de 41 unidades del producto.

Vemos que la versión base como la segunda modificación son muy similares, siendo la única diferencia el momento en el que se realiza el pedido. En la versión base se hace un pedido mensual (si es necesario), mientras que en la segunda modificación siempre que se acabe el stock del producto, este es repuesto.

Según los datos obtenidos, la versión base obtiene un coste total menor al que obtienen ambas modificaciones. Aunque queda muy claro que la primera modificación es la peor versión de las tres, la segunda obtiene unos resultados lo bastante buenos como para que se considere similar a la versión en función al coste total. Si nos fijamos en el coste de pedido, la versión base tiene menor coste de pedido y mayor coste de déficit, lo que es lógico, porque realiza menos pedidos.

La primera modificación vemos como apenas genera coste por déficit, puesto que siempre pide una cantidad de producto razonable (basada en las ventas anteriores), pero si que obtiene un elevado coste de mantenimiento.

En definitiva, ninguna de las dos modificaciones aporta resultados como para ser consideradas superiores a la versión base.