|  |  |
| --- | --- |
|  | **Pontificia Universidad CatÓlica de Chile**  **Escuela de IngenierÍa**  **DEPARTAMENTO DE INGENIERÍA INDUSTRIAL Y DE SISTEMAS**  **PROFESOR: PEDRO GAZMURI S.**  **ICS 3723 – SIMULACIÓN** |

.

Tarea N°4

Julio Adriazola Soto

11-06-2013

Índice

[1 Resumen Ejecutivo 2](#_Toc358707236)

[2 Pregunta 1: Ciclos regenerativos para variables correlacionadas 2](#_Toc358707237)

[3 Pregunta 2: Búsqueda de la mejor configuración. 3](#_Toc358707238)

[4 Pregunta 3: Optimización con OptQuest® 3](#_Toc358707239)

[4.1 Modelación y optimización del sistema 4](#_Toc358707240)

[4.1.1 Componentes 4](#_Toc358707241)

[4.1.2 Modelo en Arena 5](#_Toc358707242)

[4.1.3 Optimización del modelo: Uso de OptQuest® 6](#_Toc358707243)

[4.2 Conclusiones 11](#_Toc358707244)

[5 Anexos 12](#_Toc358707245)

Índice de Ilustraciones

[Ilustración 1: Histograma para el número de pasos por ciclo 3](#_Toc358707263)

[Ilustración 2: Histograma para el tiempo de espera en el sistema 3](#_Toc358707264)

[Ilustración 3: Configuración del sistema de atención de personas 4](#_Toc358707265)

[Ilustración 4: Parte 1 modelo en Arena 6](#_Toc358707266)

[Ilustración 5: Parte 2 modelo en Arena 7](#_Toc358707267)

[Ilustración 6: Estabilización de la media 11](#_Toc358707268)

[Ilustración 7: Estabilización de la varianza 11](#_Toc358707269)

Índice de Tablas

[Tabla 1: Mejores 10 resultados para las 400 iteraciones de la fase 1. 9](#_Toc358707446)

[Tabla 2: Resultado de aplicación de 10 réplicas adicionales a las 10 mejores soluciones 10](#_Toc358707447)

[Tabla 3: Valores para cada ciclo de regeneración 13](#_Toc358707448)

# Resumen Ejecutivo

En este documento se explicará la resolución de la Tarea 4 propuesta para el curso de simulación de la Pontificia Universidad Católica de Chile, en el año 2013.

Los temas principales a abordar son la aplicación de ciclos regenerativos para un sistema de espera en el cual existe correlación entre los tiempos de llegada, con lo cual se obtendrán estimaciones de largo plazo.

Para la pregunta 2 se cita nuevamente el problema visto en las tareas 1 y 3, pero en esta ocasión se busca evaluar 5 escenarios distintos para elegir el mejor de estos, lo cual lograremos a través de alguno de los métodos de comparación vistos en clases.

En la última pregunta se plantea un problema de optimización, en el cual existe un sistema de espera y se busca variar las tazas de atención para disminuir los costos bajo ciertas restricciones, esto se logrará usando la herramienta complementaria a Arena, OptQuest®.

# Pregunta 1: Ciclos regenerativos para variables correlacionadas

En este ejercicio se nos pide aplicar ciclos regenerativos para las variables implicadas. En un comienzo, realizamos histogramas tanto para Nj como Zj, el resultado es el siguiente:

Ilustración : Histograma para el número de pasos por ciclo

Ilustración : Histograma para el tiempo de espera en el sistema

# Pregunta 2: Búsqueda de la mejor configuración.

# Pregunta 3: Optimización con OptQuest®

El problema trata de un sistema de espera en donde cada persona será atendido en 2 servidores consecutivos, pero que seguirán un camino distinto según sean del tipo 1 o 2. El diagrama general del problema se muestra a continuación.

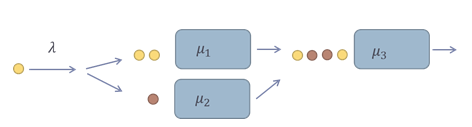


Ilustración 3: Configuración del sistema de atención de personas

Lo que se busca es encontrar las tasas , y que minimicen nuestra función de costos, la cual conocemos y se expresa como .

Sin embargo, se deben cumplir ciertas restricciones relacionadas al nivel de servicio. Dicho nivel se define como el porcentaje de personas que esperan menos de 15 minutos en las colas. Las políticas nos dicen que este indicador no debe ser inferior al 95% para las personas del tipo 1, y al 85% para las del tipo 2.

Una solución no computacional podría consistir en plantear distintos escenarios posibles y buscar el mejor de estos, o aplicar un algoritmo de optimización como el de descenso, sin embargo, estos serían de alto costo y existe la posibilidad de cometer errores humanos que empañarían los resultados. Por este motivo, usaremos el complemento de Arena, OptQuest®.

## Modelación y optimización del sistema

En una breve descripción, el modelo consta de lo siguiente.

### Componentes

Las entidades corresponden a las personas que pasan por el sistema, estas pueden ser de tipo 1 con 75% de probabilidad, y de tipo 2 con el 25%.

Por otro lado, las variables aleatorias de input son el tiempo entre llegadas, y los tiempos de atención en los servicios 1, 2 y 3, los cuales distribuyen exponencial con tasas son , , y respectivamente.

Las principales variables a considerar son las necesarias para poder hacer una optimización en OptQuest® considerando el problema presentado, lo que corresponde a las 4 tasas ya mencionadas, y 4 variables que guardan la cantidad de clientes “bien” y “mal” atendidos por cada tipo, en donde ser bien atendido significa que estuvo menos de 15 minutos en cola. Adicionalmente se utilizaron algunos atributos auxiliares –asociados a cada entidad- para obtener el tiempo que espera en cola cada persona.

### Modelo en Arena

El problema se modeló básicamente en 2 partes.

En la primera parte (Ver Ilustración 2) se crean las entidades en base a la distribución ya mencionada y posteriormente éstas pasan por un decide, de tipo *2-way by chance*, en el cual con un 25% de probabilidades se tomará la vía de arriba. Los 2 caminos funcionan de forma análoga, existe un *Assign* que setea el tipo de persona en el atributo *Tipo*, se guarda en un atributo *EntraACola* el tiempo actual de la simulación para posteriormente poder calcular el tiempo que esperó en la cola, y finalmente se setea en 0 un atributo *tiempoEsperaEnCola*, que dará la cantidad total que esperó cada persona en la cola en horas.

Posteriormente existe un *Hold* el cual liberará las entidades solo si en el proceso que le sigue hay 0 entidades atendiéndose, es decir, hace de cola para el servidor respectivo. El siguiente *Assign* hace el cálculo del tiempo que estuvo en cola la entidad y lo suma al valor actual de *tiempoEsperaEnCola*. Posteriormente pasa al proceso en donde simplemente se modeló como un *Delay* con la distribución exponencial ya mencionada. Una vez realizado esto, los caminos se unen y pasan a la segunda parte del modelo.

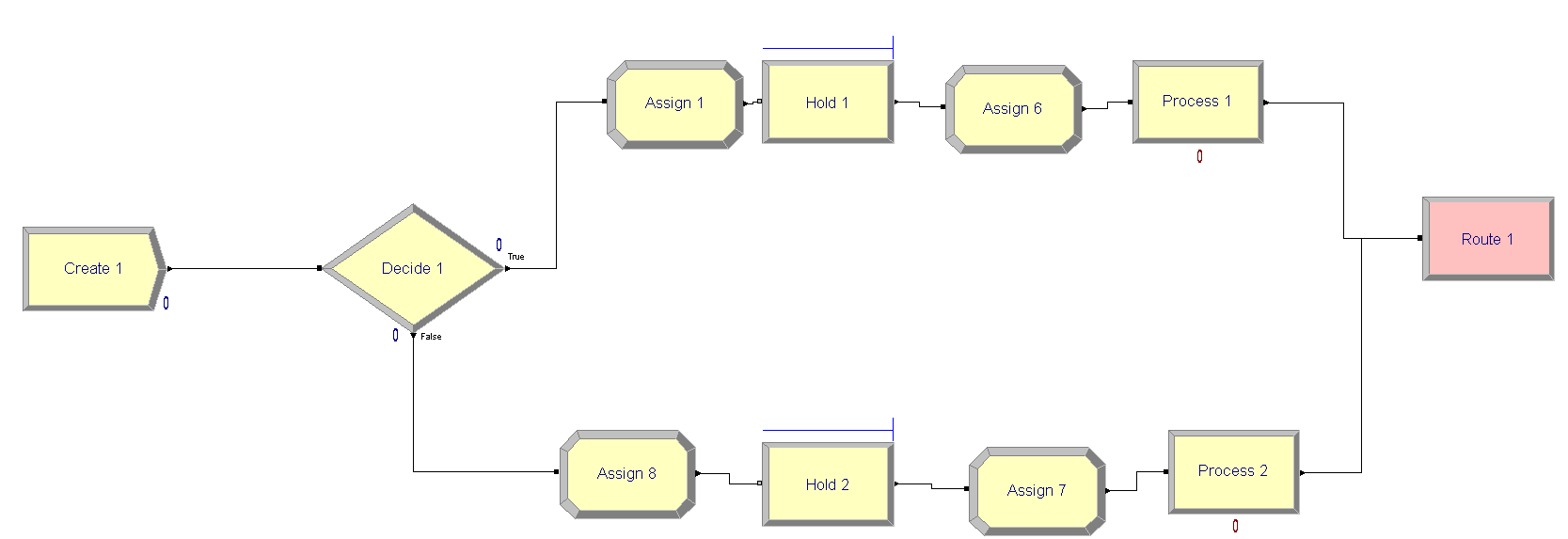


Ilustración 4: Parte 1 modelo en Arena

En la segunda parte del modelo (ver Ilustración 3) existe el mismo sistema de cola y cálculo de tiempo en cola ya mencionado para la parte 1, con lo cual viene en esta ocasión un proceso correspondiente al servidor 3. Una vez realizado esto, cada entidad pasa por un *Decide* del tipo *N-way by conditon*, en el cual se analiza cada opción posible. Por el primer camino pasan las entidades de tipo 1 cuyo tiempo total en cola fue menor a 15 minutos (0.25 horas), en el segundo camino las que son de tipo 1 y esperaron a lo menos 15 minutos. En el tercer camino pasan las entidades de tipo 2 cuyo tiempo fue menor a 15 minutos y en el resto de los casos, es decir, que sea de tipo 2 y cuya espera sea a lo menos de un cuarto de hora, toma la vía del *else*.

En cada camino se aumentará en 1 la cantidad de atendidos según tipo y según la atención brindada, es decir, aumentará en 1 la variable bienAtendidos1, malAtendidos1, bienAtendidos2 y malAtendidos2, respectivamente. Una vez realizado esto, se destruyen las entidades con un *dispose*, con lo cual podemos ahorrar recursos del computador.

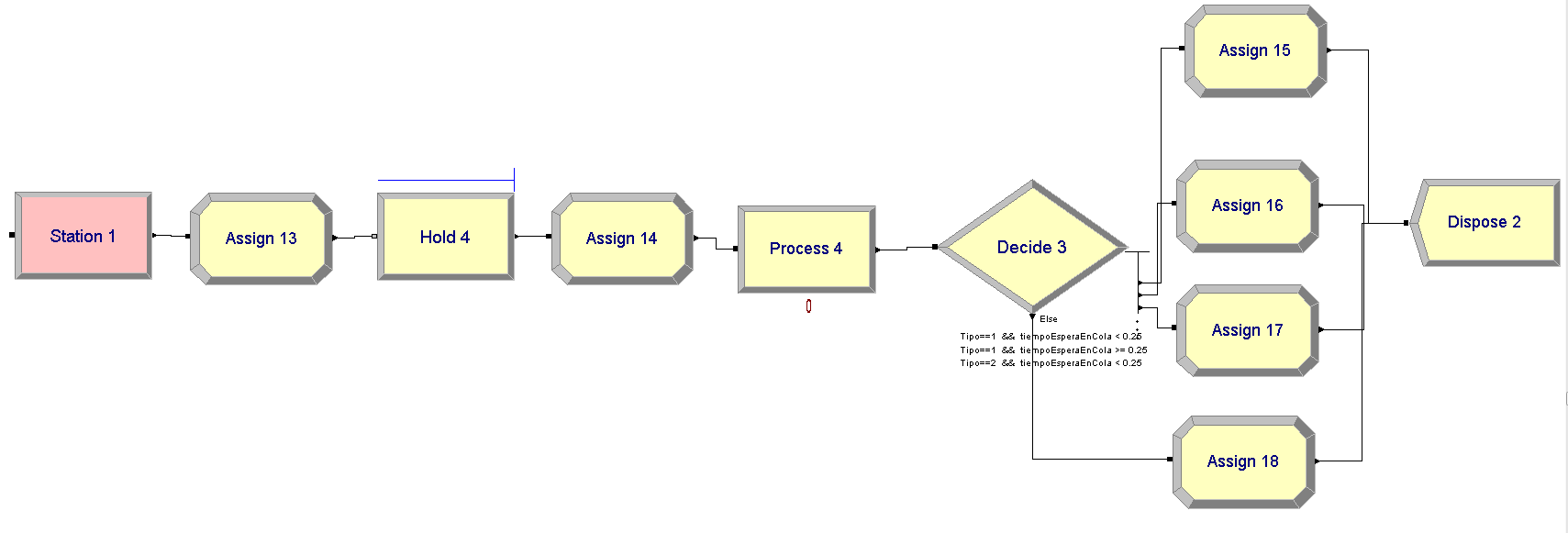


Ilustración 5: Parte 2 modelo en Arena

Adicionalmente, se estableció como unidad básica de tiempo para el modelo la Hora, y la duración de cada réplica de 12 horas.

### Optimización del modelo: Uso de OptQuest®

En resumen, queremos resolver el siguiente problema de optimización:

.

Para ello vamos al menú de Arena y seleccionamos *Tools*, y dentro de este menú *OptQuest for Arena*. Una vez realizado esto, seleccionamos la opción de *New Optimizaion.*

En este problema, seleccionaremos como *Controls* mu1, mu2 y mu3, que son los valores que deseamos variar, y para acotar el espacio de soluciones factibles, diremos que son de tipo entero, dado que a la hora de hablar de tasas tiene mayor sentido, y acotaremos sus valores entre 25 y 200 para mu1, 75 y 200 para mu2 y 100 y 200 para mu3, con valor recomendado para todos de 100, es decir, que parte con el mismo valor de la tasa de llegada. Los valores mínimos se basan en considerar a cada sistema aislado como un M/M/1, donde la tasa de atención es el equivalente a la tasa de entrada a ese sistema en particular, mientras que el valor máximo es un valor que me pareció prudente considerar.

Para nuestras *Responses* elegiremos las variables *bienAtendidos1*, *malAtendidos1*, *bienAtendidos2* y mal*Atendidos2*, que son efectivamente las que formarán parte de nuestras restricciones y querremos que Arena saque un promedio de estas sobre todas las réplicas realizadas.

Agregaremos 2 restricciones, las cuales pondremos explícitamente a continuación:

[bienAtendidos1]/([bienAtendidos1]+[malAtendidos1])>=0.95

[bienAtendidos2]/([bienAtendidos2]+[malAtendidos2])>=0.85

Estas restricciones corresponden a mantener un nivel mínimo de servicio.

Para *objectives* pondremos nuestra función objetivo que corresponde a:

45\*[mu1]+25\*[mu2]+60\*[mu3]

Por último, en *Options* se especifica que la optimización terminará solo después de 400 iteraciones, y que cada iteración será de 3 réplicas, con todo esto se echa a correr el programa.

#### Fase 1: Orden de magnitud de las variables de control

En una primera instancia, nos interesa saber cuál es el orden de magnitud que toman mu1, mu2 y mu3, para poder acotar de mejor forma el problema. Lo que hacemos es simplemente correr 50 iteraciones de 1 réplica cada una. Acotamos inicialmente las tasas por los valores ya mencionados. Los resultados se pueden revisar en el archivo *OptQuestSolutions.csv* y con estos acotaremos más aún el espacio de soluciones factibles, y los nuevos valores de mu1 estarán entre 35 y 85, para mu2 entre 90 y 130, y para mu3 entre 100 y 135. El valor sugerido será el mínimo de cada intervalo, además, agregamos las 10 mejores soluciones a las soluciones sugeridas. Considerando esto, recien aplicamos las configuraciones antes señaladas.

#### Fase 2: Resultados primera optimización

En la primera fase OptQuest® realizó las 400 iteraciones y nos quedamos solo con las 10 mejores. El resultado se puede revisar en el archivo *OptQuestSolutions2.csv*.

Las 10 mejores opciones se muestran a continuación

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Iteración | Objetivo | Nivel 1 | Nivel 2 | mu1 | mu2 | mu3 |
| 314 | 10635 | 0.975393 | 0.97755 | 40 | 99 | 106 |
| 234 | 10665 | 0.960164 | 0.981106 | 39 | 102 | 106 |
| 298 | 10730 | 0.96264 | 0.973391 | 41 | 101 | 106 |
| 194 | 10790 | 0.975488 | 0.993206 | 40 | 98 | 109 |
| 187 | 10805 | 0.953093 | 0.975492 | 40 | 101 | 108 |
| 143 | 10995 | 0.964345 | 0.976973 | 40 | 99 | 112 |
| 225 | 11025 | 0.983071 | 0.968533 | 43 | 90 | 114 |
| 85 | 11045 | 0.989013 | 0.985485 | 40 | 101 | 112 |
| 171 | 11055 | 0.997765 | 0.995846 | 41 | 90 | 116 |
| 100 | 11065 | 0.957433 | 0.980545 | 41 | 100 | 112 |
| AVG | 10881 | 0.97184 | 0.980813 | 40.5 | 98.1 | 110.1 |
| SD | 172.8165 | 0.014739 | 0.008571 | 1.080123 | 4.433459 | 3.60401 |

Tabla : Mejores 10 resultados para las 400 iteraciones de la fase 1.

Como podemos ver, los valores para mu1 se asemejan en todos los casos y están en torno a 40.5 con una desviación estándar de 1.01, es decir, el valor óptimo para mu1 está cercano al 40. Para mu2 y mu3 es más difícil hacer una conjetura de este tipo, debido a que ambas tienen una desviación estándar mayor. Por otro lado notamos que para algunos casos mu2 toma el extremo inferior que definimos para el intervalo, lo que quiere decir que pudimos haber fallado en nuestra estimación, y existe la posibilidad de que en realidad el óptimo de mu2 sea inferior a este valor, sin embargo, dado el costo de correr todo de nuevo, ignoraremos esto.

Existe un punto importante a considerar y este es que tenemos tan solo 3 réplicas para cada iteración, y cada réplica es de tan solo 12 horas, con lo cual es muy seguro que el periodo transiente tenga un gran efecto sobre la media calculada, y, por lo tanto, los valores a los que nos estamos refiriendo no representan la verdadera realidad del sistema. Puede suceder que en realidad parte de estas 10 mejores configuraciones sean no factibles, y, con una alta probabilidad, también puede suceder que hayamos dejado afuera la mejor configuración.

Dada la simplicidad del problema, podríamos establecer un modelo matemático que lo resuelva de forma mucho más exacta, pero no es de nuestro interés realizarlo. Por lo tanto, para mejorar nuestros resultados, tenemos básicamente 2 opciones, las cuales se pueden combinar sin ningún problema:

1. Considerar un largo mayor para cada réplica
2. Aumentar el número de réplicas para cada iteración

En este caso sólo consideraremos la segunda opción. Hacer esto para las 400 iteraciones implicaría realizar 400N réplicas (donde N sería el nuevo número de réplicas por iteración), lo que implica un esfuerzo 39 veces mayor de lo que implica hacerlo solo para las actuales 10 mejores configuraciones, es decir, si al aplicar N réplicas para las 10 mejores configuraciones nos demoramos 2 minutos, para las 400 iteraciones significaría una demora de 80 minutos. Si bien, con una alta probabilidad daríamos con la configuración óptima, el costo de realizar esto para todas las iteraciones es demasiado alto, por lo que aumentaremos el número de réplicas solo para estas 10 configuraciones.

#### Fase 3: Aumento de réplicas a las 10 mejores configuraciones

Una vez obtenido el resultado, escogemos la opción *Refine Solutions*, y seleccionamos la opción para hacer más réplicas. Para mejorar los resultados, haremos 10 réplicas adicionales para las 10 mejores configuraciones.

El resultado fue el siguiente:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Iteración | Factible | Objetivo | Nivel 1 | Nivel 2 | mu1 | mu2 | mu3 |
| 314 | No | 10635 | 0.87588 | 0.892617 | 40 | 99 | 106 |
| 234 | No | 10665 | 0.886008 | 0.918843 | 39 | 102 | 106 |
| 298 | No | 10730 | 0.8845 | 0.919418 | 41 | 101 | 106 |
| 194 | No | 10790 | 0.946809 | 0.964325 | 40 | 98 | 109 |
| 187 | No | 10805 | 0.878634 | 0.904436 | 40 | 101 | 108 |
| 143 | No | 10995 | 0.949665 | 0.972182 | 40 | 99 | 112 |
| **225** | **Sí** | **11025** | **0.964294** | **0.963937** | **43** | **90** | **114** |
| 85 | No | 11045 | 0.946535 | 0.976417 | 40 | 101 | 112 |
| 171 | No | 11055 | 0.947252 | 0.951856 | 41 | 90 | 116 |
| 100 | No | 11065 | 0.947365 | 0.952125 | 41 | 100 | 112 |

Tabla : Resultado de aplicación de 10 réplicas adicionales a las 10 mejores soluciones

Como podemos notar, 9 de las 10 configuraciones que eran factibles ya no lo son, lo que significa que los estándares de atención no se están cumpliendo en estas réplicas. Por otro lado, dado que la función objetivo está solo en función de las tasas, esta no cambia su valor en ningún caso.

Si analizamos el output con respecto al nivel de servicio nos damos cuenta que al avanzar en las iteraciones el valor de la media y la varianza se estabilizan.

Ilustración : Estabilización de la media

Ilustración : Estabilización de la varianza

Se esperaría que al tratarse de escenarios distintos, no existiera estabilización, sin embargo este comportamiento se puede explicar a través de la forma de operar de OptQuest®, cuyas soluciones están dentro de un espacio de soluciones factibles bien acotado, por lo que no existe la posibilidad de que una tasa tome un valor exagerado como 1000, y de esta manera, tampoco debiera existir una variación significativa en los resultados.

## Conclusiones

A partir de nuestro problema en OptQuest®, y con la información que manejamos, podemos determinar que el mejor escenario encontrado es al considerar mu1, mu2 y mu3 como 43, 90 y 114, respectivamente. Sin embargo, aún puede disminuirse más el costo, dado que, trabajamos solo con las 10 mejores soluciones factibles, pero puede suceder que en el resto de configuraciones en las que iteró OptQuest® esté el óptimo real, o incluso existe la posibilidad de que OptQuest® nunca haya considerado el óptimo teórico.

Así como vimos que algunas soluciones factibles pasan a ser no factibles, también puede suceder lo contrario, y, por lo tanto, la solución óptima podría estar entre las que salieron en la fase 2 como no factibles. La forma que escojamos para mejorar nuestros resultados implicará si o si un costo adicional, y estas son considerar un mayor número de réplicas iniciales, y/o aumentar el número de iteraciones.

# Anexos

Tabla referida en el punto 1:

|  |  |  |  |
| --- | --- | --- | --- |
| j | Bj | Nj | Zj |
| 1 | 4 | 9 | 24 |
| 2 | 13 | 17 | 73 |
| 3 | 30 | 40 | 259 |
| 4 | 70 | 104 | 701 |
| 5 | 174 | 8 | 22 |
| 6 | 182 | 2 | 3 |
| 7 | 184 | 15 | 29 |
| 8 | 199 | 4 | 10 |
| 9 | 203 | 2 | 4 |
| 10 | 205 | 11 | 22 |
| 11 | 216 | 5 | 5 |
| 12 | 221 | 81 | 649 |
| 13 | 302 | 6 | 15 |
| 14 | 308 | 3 | 6 |
| 15 | 311 | 24 | 53 |
| 16 | 335 | 5 | 16 |
| 17 | 340 | 30 | 34 |
| 18 | 370 | 164 | 1311 |
| 19 | 534 | 73 | 386 |
| 20 | 607 | 2 | 3 |
| 21 | 609 | 7 | 13 |
| 22 | 616 | 2 | 3 |
| 23 | 618 | 29 | 85 |
| 24 | 647 | 3 | 6 |
| 25 | 650 | 20 | 17 |
| 26 | 670 | 99 | 648 |
| 27 | 769 | 20 | 28 |
| 28 | 789 | 8 | 7 |
| 29 | 797 | 15 | 20 |
| 30 | 812 | 16 | 9 |
| 31 | 828 | 4 | 12 |
| 32 | 832 | 2 | 4 |
| 33 | 834 | 8 | 3 |
| 34 | 842 | 7 | 15 |
| 35 | 849 | 7 | 4 |
| 36 | 856 | 13 | 18 |
| 37 | 869 | 4 | 11 |
| 38 | 873 | 7 | 21 |
| 39 | 880 | 21 | 34 |
| 40 | 901 | 20 | 22 |
| 41 | 921 | 2 | 3 |
| 42 | 923 | 2 | 4 |
| 43 | 925 | 4 | 7 |
| 44 | 929 | 10 | 22 |

Tabla : Valores para cada ciclo de regeneración