

Actividad 6.1 Análisis de herramientas usadas para la solución de las situaciones problema.

En la elaboración de la situación problema nos enfrentamos a grandes retos y dificultades para la realización del proyecto, sin embargo gracias a herramientas tecnológicas pudimos sobrellevar todas las problemáticas que se presentaban a lo largo del camino. Existen varias herramientas que nos pudieron ayudar en la situación problema, es por eso que mencionaremos las herramientas que utilizamos y algunas alternativas.

Comunicación

Para el área de comunicación el equipo utilizó una herramienta tecnológica sumamente útil. La herramienta es la plataforma de Discord, esta plataforma te permite realizar varias acciones como mensajear, compartir documentos, compartir fotos, comunicarte por videollamada, incluso te permite streamear para que los demás miembros del equipo puedan observar lo que realizas en tiempo real. Una plataforma similar al Discord es Zoom, las dos herramientas te permiten comunicarte con tu equipo, sin embargo la gran diferencia es que en Discord tienes más funcionalidades, la eficiencia de comunicación es mayor porque no tiene un límite de tiempo, existen notificaciones dentro de la plataforma que aumentan la eficacia de Discord, además tiene shortcuts y formas sencillas de interactuar con la plataforma que te permiten tener mejor agilidad.

Trabajo colaborativo

En el aspecto de trabajo colaborativo Github nos permitió organizar y compartirnos los archivos, así como la capacidad de que todos los miembros del equipo aporten en los trabajos colaborativos. Github es tan útil que todos los cambios que se hayan realizado en los documentos se guardan de manera online para que no se pierda la información. Una herramienta similar es Google Drive o Dropbox, aunque estas dos herramientas no cuentan con la compatibilidad de los lenguajes de programación lo que las convierte en herramientas menos funcionales, Github también cuenta con shortcuts y te permite trabajar dentro de la terminal, esto

provoca que exista una mejor eficacia y agilidad en los trabajos. Se mencionó que en Github se puede trabajar desde la terminal, pero para algunos miembros podría ser difícil adaptarse a trabajar con la terminal, la ventaja que existe es que al igual que Drive o Dropbox se puede trabajar con pagina web o app así que tienen una facilidad de uso. El elemento clave de Github es que permite sincronizar carpetas en los distintos dispositivos de tu equipo, por eso Github es una herramienta eficiente.

Diagramas

Para la planeación de los DFA nos apoyamos de la herramienta Lucidchart, la función de Lucidchart es la creación de distintos diagramas de manera sencilla y en línea, nos permite trabajar de manera colaborativa con todos los integrantes del equipo en tiempo real. La herramienta que de igual manera nos pudo ayudar fue Jflap que es utilizado para simular diagramas enfocados al lenguaje de autómatas. En este caso Jflap al estar enfocada al tipo de trabajo que queríamos realizar era una opción más atractiva y eficiente, sin embargo el poder trabajar colaborativamente desde Lucidchart permite que sea más eficaz.

Herramienta de editores de texto

Para la implementación del DFA se usó Excel de apoyo, con Excel definimos los casos que se tendrían para el DFA y de esta manera poder exportarlo en un formato csv para poder utilizarlo dentro de python. Dos herramientas que son muy similares a Excel son Google sheets y Numbers, porque te permiten crear el mismo tipo de documentos. La gran diferencia es que en Excel se tienen mejor accesibilidad a todas las funciones lo que convierte a Excel en una herramienta ágil, El único inconveniente que se presenta cuando se compara con Google sheets es que no tiene la capacidad de trabajar de manera colaborativa en línea.

La documentación de los proyectos que realizamos fue mayormente hecha en word o Google docs. Ambas herramientas son editores de texto enfocados a la realización de documentos. Alternamos entre ambos debido a que Google docs lo usamos para trabajo colaborativo simultáneo en la nube y word para darle formato a los documentos de manera más precisa o si bien para ciertos documentos no tan largos. La ventaja de word son sus amplias opciones de personalización y configuración del documento. Google docs pierde en cuanto a estas opciones pero

cuenta con la ventaja de que su opción para trabajo colaborativo es rápida y sencilla de configurar a diferencia de la opción que ofrece google.

Editores de código

Para llevar a cabo nuestras actividades y la situación problema usamos el editor de texto Visual studio code. Si bien Vscod no es un IDE por sí solo, gracias a la gran cantidad de plugins que se le pueden incorporar puede pasar a ser una herramienta muy completa y similar al de un IDE como lo son Visual studio o las incluidas en el portafolio de JetBrains. La razón por la cual utilizamos Vscod se debe a que ya estamos familiarizados con el entorno, su poco consumo de recursos de la computadora y su capacidad de poder trabajar con distintos lenguajes e intérpretes gracias a la instalación de plugins. Si bien las funcionalidades de un IDE son bastante amplias, la ventaja de Vscod es su sencillez de ampliar la propia configuración del editor y sus capacidades al gusto propio además de la carga más ligera a los recursos de la máquina. Igualmente el uso del editor y los plugins es completamente gratis así que fue accesible para todos los integrantes del equipo. Implementar código de python en VsCode es realmente sencillo y cómodo gracias a la extensión de capacidades del editor con plugins de debugeo, corrección de código, testeo, entre otras funcionalidades. Claro que comparándolo con Pycharm, el cual es un IDE enfocado al desarrollo en python, las funcionalidades de vscod no son tan complejas. De acuerdo con Osama Orshad(s.f) las características en las que pycharm destaca o de plano soporta contrario a vscod para el desarrollo en python son: “debugeo remoto, búsqueda y filtrado global, soporte de proyectos Django, experiencia de usuario consistente entre software de la familia jetbrains y emulación vim.” Si bien estas características son buenas, no las consideramos necesarias para el desarrollo de nuestro proyectos.

Lenguajes de programación, marcado y diseño

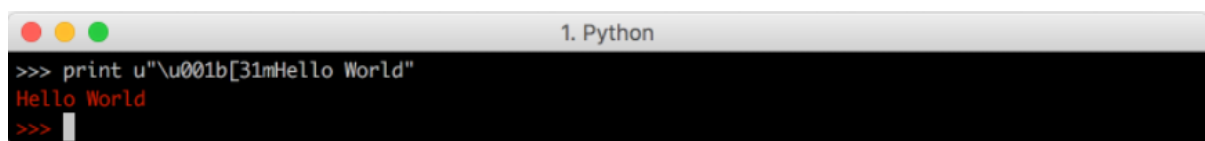
El principal lenguaje de programación que usamos dentro de nuestra solución para la situación problema fue Python. Lo usamos no solo como el código base para la implementación de los algoritmos de nuestras soluciones, sino también como backend dentro de nuestra página web donde tenemos los diferentes algoritmos que se implementaron como parte de la materia para que el público pueda usarlos sin necesidad de descargar ninguna herramienta. El uso de Python dentro de los algoritmos para las soluciones del reto fue debido a que: en primer lugar, los 3 integrantes ya teníamos experiencia usando el lenguaje, y en segundo lugar es un lenguaje fácil de programar, con

un desarrollo rápido lo que nos permitía integrar las soluciones en cuestión de horas. La desventaja más grande de Python frente a otros lenguajes de programación es que al ser un lenguaje interpretado, y no compilado, tarda más en su ejecución que algunos otros (como C++). Sin embargo, debido a que el problema que se nos presentaba no importaba si el programa tardaba unas décimas de segundo más en ejecutarse Python fue nuestra opción.

Parar el backend de nuestra página Python fue la opción debido a la simplicidad de la librería flask. Esta librería te permite con Python crear las funciones básicas de un servidor, administrar las rutas, y las funciones que se deben correr en cada una. Además, es de muy fácil integración con todo lo de Python que ya teníamos hecho como la parte de trabajar con archivos. Flask nos permitió de manera muy rápida crear las funciones básicas del servidor, además de administrar todas las páginas que queríamos mostrar muy fácil. Otra opción que pudimos usar era JavaScript con su librería express. Sin embargo, como el la programación de nuestro algoritmo esta sobre Python es mucho más simple hacer la integración de dos programas que corren sobre el mismo lenguaje de programación. Además de que como se pretendía subir a un servidor en la nube, es más simple y rápido configurarlo solo para un lenguaje. Respecto a las actualizaciones del lenguaje Python podemos decir que es una comunidad activa, con actualizaciones cada cierto tiempo. Sin embargo, todas las actualizaciones del lenguaje dentro de la versión 3.X son compatibles, por lo que no hay ningún problema con nuestro código o con versiones más antiguas. Si se tuviera que mencionar una actualización importante a Python fue la 3.9, que introdujo un nuevo operador (`:=`), pero no la usamos dentro de nuestro código por lo que no hay ningún problema.

En el primer avance de la situación problema donde realizamos el autómata determinista finito decidimos proveer la salida del programa donde clasificaba las tokens en una página web. Igualmente hosteamos el programa en un servidor gratuito mostrándolo en una página web sencilla. Hicimos uso del lenguaje de marcado HTML y para el diseño CSS. En este primer avance no fue muy complejo su uso y fue más complejo el back end hecho con python mencionado anteriormente. Consideramos usarlo previo a la entrega siguiente porque al hostear el programa en una página web le facilitamos al usuario el uso del mismo y le ahorramos que tuviera que seguir una guía larga de instalación para su uso. Esto igualmente permite su uso en cualquier sistema operativo ahorrandonos tener que adaptarlo para diferentes SO. Una vez que pasamos a la entrega del resaltador léxico fue donde la implementación de HTML y CSS se volvió más compleja. Para implementar esto hicimos que el output de nuestro programa hecho en python tuviera un formato de html así al

generar la página de resultado solo tuviera que interpretar el archivo .html el navegador y directamente las tokens eran coloreadas con su respectivo color. Si bien esta implementación nos pareció buena igualmente se puede realizar de una manera distinta. Otra opción para mostrar un documento y que las tokens aparezcan con sus respectivos colores pudiera ser desde la misma terminal. Usando códigos de escape ANSI podemos darle instrucciones a la terminal para que imprima texto con cierto color. Dependiendo del terminal es el número de colores disponibles. La gran mayoría si no es que todas soportan 8 colores, varias otras soportan 16 y algunas hasta 256 colores. Dependiendo del número de tokens y los usuarios que vayan a usar el programa deberíamos tomar estas limitaciones en cuenta para la implementación. Igualmente estos códigos de escape ANSI son aceptados por terminales de sistemas basados en UNIX. Para aprovecharlos en por ejemplo la terminal de windows se tendría que cambiar la implementación. Aquí un ejemplo de cómo se vería imprimir a terminal un string usando python. `print u"\u001b[31mHelloWorld"` donde el output sería el siguiente:



```
>>> print u"\u001b[31mHello World"
Hello World
>>> 
```

Tomando esto en cuenta facilmente podríamos cambiar el programa a que su output sea a consola y asignamos un código de color ANSI a cada token. Si bien la implementación es fácil se tienen que tomar en cuenta muchas consideraciones y limitaciones respecto a la forma en que se vería en las máquinas de diferentes usuarios. Para saber más de los códigos de escape ANSI refiérase a la referencia de Haoyi. (2016, 2 julio)

En la introducción a la segunda situación problema se nos presentó el tema de programación paralela en conjunto con el lenguaje de programación Java. Si bien la implementación de programación paralela nos es exclusiva a un lenguaje, ciertas librerías hacen que su implementación sea posible en unos lenguajes mientras que en otros como java su uso está soportado desde el propio lenguaje(Escuela ingeniería Segovia, s.f.). En el caso de java solo es necesario usar la clase **thread** y sobrescribir el método run. La forma alternativa de implementar hilos que es en lo que se basa la programación paralela es con la interfaz runnable en los casos donde se desea hacer herencia múltiple la cual no existe en java. En la entrega final de la segunda situación problema tendremos que implementar nuestro resaltador léxico con programación paralela. Dicha implementación podrá ser escribiendo el código de nuevo en python o ejecutando el código de generación de hilos directamente de los scripts en java con ayuda de alguna librería de python.

Servidor en la nube

Para que nuestro proyecto se pudiera correr dentro de cualquier computadora sin tener que estar pensando en las herramientas que deberían de tener instaladas o en los requerimientos de librerías específicas en cada uno de los algoritmos lo que hicimos fue hacer el deploy dentro de un servidor de heroku. Se eligió esta herramienta debido a que: en primer lugar es gratuita por lo que no debíamos de hacer ningún pago por su uso, en segundo lugar es compatible con todas las herramientas que estábamos usando como GitHub o Python, y en tercer lugar pero no menos importante es que es de fácil configuración. Esto debido a que solo requiere de 3 archivos de configuración para que pueda hacer toda la instalación de los programas y en menos de 1 hora tener una página web funcional con los archivos que ya tienes en tu repositorio. Otra herramienta que pudimos haber usado es AWS, sin embargo para usarla se tiene que dar de alta una cuenta de pago para habilitar el free-tier, además de que no es tan fácil de configurar como heroku. Requiere de más *expertise* para poderse usar. Es por eso que heroku era la mejor opción para nuestro proyecto y fue la opción que se implementó. Respecto a las actualizaciones de plataforma podemos decir que es una plataforma activa, que cada cierto tiempo libera actualizaciones respecto al uso de su plataforma y de las herramientas que pone al alcance de los programadores. Sin embargo, en los últimos meses no ha habido actualizaciones que transformen en gran manera la plataforma por lo que sigue soportando versiones anteriores de algunos lenguajes de código, además de no tener que hacer revisión de versiones de parte del programador.

Programación paralela

Retomando el concepto mencionado anteriormente de programación paralela, esta forma de implementación de código tiene sus ventajas y desventajas. Primeramente la técnica de la computación paralela se puede definir como “el uso de múltiples recursos computacionales para resolver un problema. Se distingue de la computación secuencial en que varias operaciones pueden ocurrir simultáneamente.”(Bernal,F. s.f.) Dicha técnica abre las puertas a implementaciones que no serían óptimas ni en recursos ni en tiempo al ser implementadas de manera secuencial. A partir de la idea de “divide y vencerás” se pueden ejecutar soluciones de orden mayor y con complejidad mayor. En cuanto a ventajas de esta técnica se tenemos las siguientes: resolución de problemas en tiempos razonables, ejecución de código más rápido (acelerado), ejecución de varias instrucciones de manera simultánea, división de tareas en partes independientes, buena escalabilidad y mejor balance entre rendimiento y costo que la computación secuencial.(Bernal,F. s.f.) Por el otro lado esta técnica no es beneficiosa en todos los casos y cuenta con las siguientes

desventajas: “mayor consumo de energía, dificultad de implementación, retardos por comunicación entre componentes, Número de componentes usados es directamente proporcional a los fallos potenciales, costos de producción y mantenimiento y condiciones de carrera que pueden llevar a la corrupción de datos si son mal sincronizados.” (Bernal,F. s.f.) Tomando en cuenta lo anterior consideramos que depende mucho del análisis del alcance y la complejidad de un programa el optar por implementar esta técnica además de tomar en cuenta la posibilidad de que una máquina pueda hacer uso óptimo del multiprocesamiento. En el análisis que hicimos al comparar el código de suma de números primos nos dimos cuenta que el uso de hilos tiene un límite en cuanto a su beneficio en menor tiempo de ejecución. Igualmente nos dimos cuenta que implementarlo es más difícil que de manera secuencial. El usar más hilos no siempre es inversamente proporcional al tiempo de ejecución. Igualmente la sincronización de los hilos tiene que estar implementada de manera correcta para evitar riesgos de corrupción.

Expresiones regulares

Un lenguaje regular es un tipo de lenguaje formal que puede ser descrito por una expresión regular, y que por ser el lenguaje formal más simple tiene que ser generado por los operadores de unión, concatenación o * como operador de repetición un número infinito de veces. Podemos usar expresiones regulares, como lo hicimos en actividades, para encontrar lenguajes regulares en un alfabeto dado. Esta herramienta fue crucial a la hora de analizar las soluciones que podíamos implementar. Sin embargo, debido a que en el primer código usamos las tablas de transición y ya teníamos todo ese código implementado se siguieron usando las mismas funciones. De esta manera se acelera la implementación del código, además de que era más sencillo integrarlo con todo el backend y las demás funciones que ya se tenían desarrolladas. Un lenguaje regular puede ser reconocido como máquina de Turing (de sólo lectura), un autómata de pila y un autómata finito.

Autómatas Determinísticos Finitos

La creación de un Autómata Determinístico Finito para la situación problema fue esencial, ya que la problemática inicial era identificar los distintos token que un usuario podría ingresar dentro del programa. La función del ADF dentro de nuestro proyecto es reconocer e interactuar dependiendo el estado que se ingrese, es por eso que gracias a los ADF podemos clasificar lo que un usuario ingresa dentro del programa para después darle una salida en específico. La manera en que se puede verificar que usamos el ADF es cuando se le intenta agregar un caso vacío, porque no se permite un caso vacío dentro de los ADF. Cuando se contempló utilizar esta técnica para la solución de la situación problema fácilmente se pudo simular dentro de diagramas y agregar los casos dentro de una hoja de

excel para después programarlo dentro de python. La razón por la que no se utilizó un autómatas no determinístico finito es que el ADF es más rápido y puede ser más grande que el ANDF.

Conclusión

Si bien nos sentimos cómodos con las diferentes herramientas, técnicas y conceptos que usamos en el desarrollo de actividades y la situación problema, es importante que no nos quedemos en que son las únicas. Las herramientas que usamos tienen ventajas como desventajas y el valioso poder reconocerlas y optar por usar otras herramientas en los casos cuando ya deja de beneficiarnos el uso de las mismas. La habilidad de cambiar entre dichas herramientas puede aumentar la productividad del desarrollo del proyecto y además nos da conocimientos valiosos de todo el entorno de desarrollo de un proyecto. De la misma forma, conocer cómo las herramientas que usamos se van actualizando es muy importante ya que esto permite poder aprovechar el potencial de la misma al 100%. Con esto dicho, cuando ocurra que salgan nuevas herramientas al mercado podremos tener el conocimiento y el juicio para optar por cambiarnos de entorno de desarrollo. Vivimos en un mundo donde el avance tecnológico aumenta en gran medida y las herramientas cada vez se vuelven más útiles e indispensables para lograr tareas complejas, es necesario siempre estar actualizandonos para no quedar obsoletos.

Referencias

Osama Orshad, M. (s. f.). *VS Code versus PyCharm: the Smackdown*.

Arbisoft.

<https://arbisoft.com/blog/vs-code-versus-pycharm-the-smackdown/>

Escuela ingeniería Segovia. (s. f.). *HILOS (THREADS) EN JAVA*. Inf5g.

<https://www.infor.uva.es/~fdiaz/sd/doc/hilos>

Haoyi. (2016, 2 julio). *Build your own Command Line with ANSI escape codes*. Haoyi's programming blog.
<https://www.lihaoyi.com/post/BuildyourOwnCommandLinewithANSIescapecodes.html#colors>

Aguila,J(2004).Automatas Finitos no deterministas.Universidad de

Magallanes.Recuperado de:

https://kataix.umag.cl/~jaguila/Compilers/T04_AFN.pdf

Bernal, F., Albarracin, C., & Gaona, J. (s. f.). *Programación Paralela*.

Recuperado de

http://ferestrepoca.github.io/paradigmas-de-programacion/paralela/paralela_teoría/index.html#four

GHD.(2007).Autómatas finitos deterministas.GHD.Recuperado

de:<https://www.institucional.frc.utn.edu.ar/sistemas/ghd/T-M-AFD.htm>

Hopcroft, J. E., Motwani, R., & Ullman, J. D. (2008). *Introducción a la teoría de*

autómatas, lenguajes y computación(Tercera edición.). Pearson Educación.