

Instituto Tecnológico y de Estudios Superiores de Monterrey

Campus Monterrey

School of Engineering and Sciences



**Comparison between Single-Objective and Multi-Objective Algorithms
for the creation of stable student groups.**

A thesis presented by

Miguel Angel Bravo Vidales (Student)

Submitted to the
School of Engineering and Sciences
in partial fulfillment of the requirements for the degree of

Master of Science

in

Computer Science

Monterrey, Nuevo León, May, 2015

Instituto Tecnológico y de Estudios Superiores de Monterrey

Campus Monterrey

School of Engineering and Sciences

The committee members, hereby, certify that have read the thesis presented by Miguel Angel Bravo Vidales (Student) and that it is fully adequate in scope and quality as a partial requirement for the degree of Master of Science in Computer Science.

Dr. Name-Last-Name
Tecnológico de Monterrey
Principal Advisor

Committee member A's name
Committee member A's institution
Committee Member

Committee member B's name
Committee member B's institution
Committee Member

Dr. Jorge Welti Chanes
Associate Dean of Graduate Studies
School of Engineering and Sciences

Monterrey, Nuevo León, May, 2015

Declaration of Authorship

I, Miguel Angel Bravo Vidales (Student), declare that this thesis titled, "Comparison between Single-Objective and Multi-Objective Algorithms for the creation of stable student groups." and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this dissertation is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Miguel Angel Bravo Vidales (Student)
Monterrey, Nuevo León, May, 2015

©2015 by Miguel Angel Bravo Vidales (Student)
All Rights Reserved

Dedication

(Only 1 page) Thanks for all your unconditional confidence, support, patience, and encouragement. You were my main motivation for pushing through this work.

Acknowledgements

(1 suggested page, no more than 2 pages) I would like to express my deepest gratitude to all those who have been side by side with me, along the long, but also short hours (paradox?), at night.

Do not forget the Tecnológico de Monterrey support on tuition and CONACyT with the support for living.

Comparison between Single-Objective and Multi-Objective Algorithms for the creation of stable student groups.

por

Miguel Angel Bravo Vidales (Student)

Resumen

En el siguiente documento es presentada una propuesta de tesis para la obtención del grado de Maestría en Ciencias Computacionales y trata de resolver un problema de agrupar distintos usuarios para la práctica de idiomas extranjeros por medio de su nivel de experiencia y sus temas de conversación de interés, entre otros factores. Estamos considerando la existencia de “cuartos” donde los usuarios practican el idioma en cuestión, pero habiendo distintas maneras posibles de agrupar usuarios en un momento dado, también hay unas formas mejores que otras, de acuerdo con criterios predefinidos. Para esto se abordan dos posibles escenarios, en el primero se aborda el problema como uno de optimización mono-objetivo u de optimización general donde se pretende maximizar los criterios antes mencionados tomándose como pesos de una misma función. El segundo escenario es abordar el problema como uno de optimización multiobjetivo donde se realizarán pruebas con distintos algoritmos genéticos de los ya existentes para resolver este problema. Finalmente se harán distintas comparaciones entre la implementación del problema mono-objetivo con respecto a los resultados obtenidos de las pruebas usando algoritmos multi-objetivo a través de una herramienta conocida como jMetal que implementa dentro de sí algunos de los algoritmos genéticos más comunes, además de algunas de las herramientas necesarias para medir el rendimiento tanto de los algoritmos para resolver problemas mono-objetivo como para resolver problemas multiobjetivo. Teniendo en mente que el resultado probablemente tienda a ser mejor con una solución multiobjetivo, también se determinará cual de estos algoritmos resulta tener un mejor desempeño para este problema en específico.

El documento se conforma de la definición del problema donde se habla tanto del origen del problema como su especificación técnica para resolverlo, después se listan varios objetivos que buscan resolver el problema especificado. En la siguiente sección se habla de la hipótesis y las preguntas que pretende contestar en caso de ser acertada. Después en el marco teórico se especifican algunas de las herramientas, metodologías y trabajo relacionado que puede ser usado para llevar a cabo la investigación, En la sección de metodología se describen a más detalle los pasos que se llevan a cabo para llevar a cabo la experimentación y cómo se evaluarán los resultados y finalmente en el plan de trabajo se describe brevemente los tiempos necesarios para cubrir los objetivos ya mencionados.

List of Figures

List of Tables

Contents

Chapter 1

Introducción

1.1 Contexto de la Investigación

A medida que el mundo se vuelve cada vez más globalizado el contenido producido en distintos idiomas diferentes al materno aumenta, en especial el que se encuentra escrito en el idioma Inglés que resulta ser el dominante a nivel mundial. Las necesidades de aprender este idioma aumentan de manera radical a medida que el tiempo avanza y los métodos tradicionales de enseñanza no dan abasto para que se pueda enseñar a toda la población que requiere aprender, por lo que surge la necesidad de crear alternativas que resulten más accesibles y óptimas para usuarios que a menudo se encuentran ocupados por un ritmo de vida muy activo y por lo tanto tienen poca flexibilidad de horario como para practicarlo entre sus conocidos.

Por lo que una de las soluciones propuestas consiste en un sistema en el cual usuarios que no se conozcan entre sí sean agrupados según sus temas de interés, esto para practicar el uso del lenguaje conforme teniendo conversaciones sobre estos intereses y a un nivel de experiencia similar para que todos los participantes que se encuentran en un determinado momento puedan entender y participar. Lo que se pretende desarrollar como parte de esta tesis es la forma de agrupar los usuarios en los distintos grupos virtuales que se encuentran dentro de la plataforma.

La intención es buscar la manera más óptima de poder distribuir a los nuevos usuarios en estos grupos para que puedan mejorar su uso del lenguaje y no resulte frustrante ya sea por la dificultad inalcanzable o por que los temas no resulten de su agrado. Hay diversos aspectos que pueden hacer que una agrupación sea mejor que otra: por ejemplo, el nivel de dominio del lenguaje debe ser relativamente compatible, o sea que no se pueden poner usuarios muy avanzados junto con principiantes, pues el usuario avanzado se aburre con las dificultades del principiante, y este último no entiende lo que dicen los avanzados. Adicionalmente, se busca que los intereses de los usuarios sean compatibles, para poder generar conversaciones de interés para todos. Por ejemplo, algunos usuarios pueden tener los deportes entre sus temas de interés (y aún dentro de los deportes hay unos de fútbol, otros de natación, etc.). Otro criterio más es el estilo de intervención de cada usuario, el cual puede ser más o menos activo y más o menos tímido. Así, si se ponen muchos usuarios tímidos y pasivos en un cuarto, difícilmente

se va a dar una dinámica ágil que haga la sesión entretenida. Es por esto que se propone abordar el problema como uno de optimización multiobjetivo donde los temas de conversación y el nivel de experiencia del usuario resultan los objetivos que se pretenden optimizar. Para ello se planea implementar una solución mono-objetivo tomando distintos criterios como el nivel de experiencia del usuario, así como sus temas de interés, entre otros para tomarlos como pesos en una función mono-objetivo. Después se hará una comparación de los resultados obtenidos con distintos algoritmos genéticos tomando como base la solución mono-objetivo y comparándola con cada uno de los resultados de la solución multiobjetivo. Se cree que haciendo uso de una estrategia multiobjetivo la experiencia de los usuarios mejorará de manera significativa comparándola respecto a los métodos de enseñanza y práctica de idiomas extranjeros ya existentes.

1.2 Definición del Problema

El problema se define como una variación del Problema de compañeros de Habitación (Stable Roommate Problem SRP) Problem[?]. Sólo que en este caso los individuos no conocen entre sí, por lo que no tienen una preferencia real, sin embargo ésta trata de ser predecida basándose en la similitud del nivel de experiencia en el lenguaje extranjero del usuario y los intereses en común que puedan compartirse con el resto del grupo, asumiendo que estos dos factores afectan directamente la preferencia del usuario para pertenecer al grupo.

Además de esto, también se consideran como restricciones el tamaño del grupo y los estilos de participación para beneficio del resto del sistema, ya que si bien una persona puede preferir pertenecer a un grupo que tenga más en común con sí, su forma de participar puede alterar el comportamiento del resto del grupo, por ejemplo, un usuario extrovertido que participa demasiado en una conversación puede evitar que el resto del grupo participe y por otro lado un grupo en el que nadie participa necesita de un usuario extrovertido que inicie las conversaciones.

De la misma forma el tamaño del grupo se define de forma dinámica para obtener una participación homogénea entre sus integrantes, y se define de forma dinámica ya que es posible que la ausencia o presencia de algunos usuarios puede afectar el desempeño del grupo a pesar de que su tamaño no se considere ideal.

Ésta búsqueda tiene como objetivo obtener la solución más estable posible con respecto a las preferencias predecidas de los usuarios dentro de un espacio de tiempo razonable, y que dentro de cada uno de los grupos exista una participación homogénea para optimizar la práctica del idioma.

A pesar de que en la literatura existen muchos ejemplos y variaciones de SRP, por ejemplo [ref.], actualmente no existe una caracterización que se adapte directamente al problema que definido, por lo que se desconoce qué tipo de optimización resulta más conveniente, y tomando en cuenta que el tamaño de los grupos es variable se le considera como un problem

NP-completo[ref.], por lo que resulta relevante la búsqueda de una solución de distintas formas. Principalmente comparando entre algoritmos mono-objetivo (por medio de una función compuesta que considere el tamaño de los grupos, el nivel de experiencia, los intereses y el estilo de participación del usuario) y multiobjetivo que considere cada una de estas funciones de forma individual.

1.3 Objetivos

El objetivo de la investigación es comparar distintas estrategias mono-objetivo y multi-objetivo con respecto a la calidad de las soluciones que se generan y el tiempo para obtener este resultado.

Los objetivos específicos que esto conlleva se mencionan a continuación:

1. Los datos para la experimentación serán artificiales, debido a que no se cuentan con datos con las especificaciones requeridas, por lo que es necesario buscar bases de datos públicas que puedan servir para hacer pruebas, con el afán de que se acerquen lo más posible a datos reales.
2. Definir los atributos y parámetros relevantes para formular las funciones mono-objetivo y multiobjetivo, para la función mono-objetivo se establecerá un peso para cada atributo.
3. Implementar estrategias mono-objetivo para generar soluciones del problema, los cuales serán la base para poder comparar las soluciones multiobjetivo que se realicen.
4. Implementar y probar distintas estrategias multiobjetivo y compararlos con los resultados obtenidos con las estrategias mono-objetivo.
5. Revisar los resultados y realizar un reporte indicando si se llegó a un mejor resultado con una estrategia multi-objetivo y cuál fue el algoritmo que tuvo mejores resultados.

1.4 Hipótesis

La hipótesis consiste en que un método multiobjetivo puede encontrar soluciones de mejor calidad y en un tiempo menor, comparándola con una solución mono-objetivo.

El proyecto busca contestar las siguientes preguntas:

- ¿Existen soluciones que estén dentro de los máximos locales en el caso del problema mono-objetivo?
- ¿Los modelos mono-objetivo que garantizan una solución pertenecen al frente de pareto?
- ¿Qué tipo de algoritmo entrega resultados de mayor calidad con respecto al problema?
- ¿Resulta mejor usar una estrategia mono-objetivo o multi-objetivo?

Chapter 2

Antecedentes

2.1 Optimization

2.2 Single-Objective Optimization

La optimización mono-objetivo o optimización global es un tipo de problemas que consiste en encontrar el mínimo o el máximo global de una función, ya que en una función no lineal pueden haber mínimos locales, pero un mínimo global es más difícil de encontrar, también es importante tomar en cuenta que se considera un dominio de soluciones y otras limitantes, por lo que en general la solución más obvia suele ser realizar una búsqueda exhaustiva probando cada uno de los valores del dominio, sin embargo esta estrategia puede llegar a ser bastante ineficiente. En los problemas no lineales una función puede contener un número muy largo de máximos y mínimos, a estos se les denominan mínimos o máximos locales, y para encontrarlos basta con usar un método local de optimización clásico.[1], [7], [8]

Los algoritmos para encontrar un mínimo global comprenden métodos exactos y heurísticos. Dentro de los algoritmos exactos existen La búsqueda exhaustiva, búsquedas bayesianas, aproximaciones sucesivas y algunos algoritmos estocásticos como el muestreo de Monte-Carlo en el cual una simulación aleatoria es usada para encontrar una solución aproximada, Tuneleo estocástico, etc.[7]

2.3 Multi-Objective Optimization

La optimización multiobjetivo es un área del análisis de decisión de múltiples criterios que se encuentra directamente relacionada con los problemas de optimización matemática, que involucra más de una función objetivo para que sea optimizada simultáneamente. [4]

Estas soluciones pueden tener distintos criterios o características para ser consideradas opciones óptimas, por lo que se les denominan soluciones de pareto. Una solución Óptima de Pareto se define como una solución cuyas funciones objetivo pueden ser mejoradas sin degradar el resto de las funciones. Se refiere a una solución no dominada en el espacio de

criterio del problema.

Para los problemas de optimización multiobjetivo no triviales no existe una solución ideal óptima, sino que existen distintas soluciones que optimizan simultáneamente cada objetivo, en cuyo caso se dice que las funciones son contradictorias y existe un número de soluciones óptimas de Pareto. Todas las soluciones que son óptimas por Pareto se dice que son igualmente buenas para los objetivos que se hayan determinado.[4], [5]

Usualmente la idea de resolver un problema multiobjetivo es entendido como ayudar a un tomador de decisiones a considerar múltiples objetivos simultáneamente y encontrar una solución óptima de Pareto que a su discreción sea más adecuada. Por lo que el método para encontrar la solución requiere de un involucramiento del tomador de decisiones y es influenciado fuertemente por sus preferencias. Usualmente en este tipo de problemas solo se toma en cuenta un sólo tomador de decisiones pero también existe la posibilidad de múltiples tomadores de decisiones. [9]

2.4 Dominance

2.5 Evolutive Algorithms

Los algoritmos evolutivos forman parte de la computación evolutiva, son algoritmos basados en una población meta-heurísticos para resolver problemas de optimización. Un algoritmo evolutivo usa mecanismos inspirados en la evolución biológica como pueden ser la reproducción, mutación, recombinación y selección, las soluciones candidatas a la optimización forman parte de una población y una función de ajuste determina la calidad de cada una de estas soluciones. Este ajuste toma parte después de distintas iteraciones o generaciones en las que se busca acercarse lo más posible a la función objetivo.[10]

Usualmente este tipo de algoritmos tiene un buen desempeño, ya que no se encuentran influenciados directamente por una función de ajuste, la única que interacción que tienen con ésta resulta ser la de un filtro para definir las poblaciones subsecuentes. El uso de estos algoritmos para el modelado de la evolución biológica se limita únicamente a la exploración de micro-evolución o la determinación de sistemas de procesos celulares, por lo que suelen ser más comúnmente usados en problemas donde la complejidad computacional resulta un factor prohibitivo. De hecho la complejidad computacional se deriva directamente de la función de ajuste. La aproximación a esta función es una de las soluciones que supera esta dificultad. [11]

En este proceso hay dos fuerzas fundamentales que forman la base de los sistemas evolutivos:

- Los operadores de variación (que son de recombinación y mutación) crean la diversidad necesaria y por lo tanto facilitan la posibilidad de encontrar nuevas soluciones.
- La selección actúa como una fuerza que empuja la calidad.

Los algoritmos evolutivos poseen un número de características que pueden ayudar a posicionarse dentro de las familias de métodos de generación y prueba:

- Son basados en poblaciones, esto significa que procesan una cantidad variable de soluciones candidatas de forma simultánea.
- Usualmente usan una recombinación para mezclar la información de más candidatos en soluciones nuevas.
- Estos algoritmos son estocásticos.

Un algoritmo evolutivo generalmente cuenta con los siguientes componentes:

- Representación (definición de las soluciones)
- Función de evaluación (o función de ajuste)
- Población
- Selección de padres
- Operaciones de variación, recombinación y mutación
- Mecanismo de selección de sobrevivientes

Cada uno de estos componente se deben de ser especificados para definir un algoritmo en particular, por lo tanto la inicialización del algoritmo tanto como la condición de terminación deben de ser definidas también.[11]

A continuación se describen los pasos que lleva a cabo un algoritmo evolutivo comúnmente:

1. **Inicialización:** Se generan distintas soluciones de forma aleatoria
2. **Evaluación:** Cada una de las soluciones pasa a ser evaluada por la función de ajuste y se determina que tanto se acerca a su valor óptimo
3. **Terminación:** Se determina si se alcanzó el objetivo con alguno de los candidatos o si se sobrepasó el número de generaciones límite que se estableció previamente.
4. **Selección:** Se selecciona las soluciones que "sobreviven" y pasan a la siguiente generación
5. **Variación:** este paso puede darse por medio de del paso de "cromosomas" o características de las soluciones que sobrevivieron en el paso anterior para combinarse entre sí y dar nuevas soluciones "hijas", el otro método es la mutación, donde una o varias de las características se modifica de manera aleatoria para introducir un nuevo valor a la población

Este ciclo se repite dando paso cada vez a nuevas generaciones que cada vez se ajustan más a la función de evaluación hasta que se alcance el resultado deseado determinado por el paso de Terminación.[10]

2.6 Single-Objective Algorithms

2.6.1 Local Search

Also referred as Hill Climbing in maximization problems and Steepest Descent in minimization problems, the definition of this algorithm is taken from [doi.org/10.1016/j.ins.2013.02.041] and in there is Stutzle in his PhD dissertation who actually does not take credit for this approach and instead mentions specific instances of the algorithm included in the literature.

Local Search is a single objective optimization meta-heuristic based on the idea of generating random starting solutions, then in each generation LS generates the starting solution for the next iteration by perturbing the local optimum found in the current iteration.

Its important to mention that the perturbing mechanism is an important feature of LS, since a weak perturbation may not be enough to escape the local optimum, and on the other hand a too strong perturbation may make the algorithm too similar to a multistart local search with randomly generated starting solutions.

A pseudocode for this algorithm can be seen in the figure [Insert figure].

2.6.2 Genetic Algorithms

This is probably the most known kind of metaheuristic for optimization, it was developed in the early 70's in Michigan with John Hoolland and some of his students who were doing research in adaptive systems at the time. [https://doi.org/10.1016/j.ins.2013.02.041]

The definition can be very generic and most of its parts are usually implemented differently according to each problem. Its essential parts include a representation of the solution known as Chromosome, a selection strategy a type of cross-over and a mutation operator.

The Chromosome is a representation of the solution and usually includes inside of it a string of genes, most commonly represented as a fixed binary string, although it can also be represented by any kind of value for combinatorial problems. Mutation is therefore done by changing a single gene to a different value, performing a bit-flip in the case of the binary representation, for example.

Crossover is a recombination of the genes, usually done with 2 individuals, which exchange their parts to create a new individual, this follows different strategies such as the n-point crossover in which a point will determine which parts will be inherited from one of its parent and which from the other. An external parameter known as pc (crossover rate) indicates the probability of each individual to undergo crossover, this value ranges from 0.6 to 1.0 according to [13 in https://doi.org/10.1016/j.ins.2013.02.041].

Individuals that can produce offspring are then chosen using a selection strategy after the fitness of each individual has been evaluated. The most common selection scheme is the roulette-wheel selection, but other types of selections such as tournament selection and ranking selection are also common.

After the crossover the individuals are subjected to mutation. This mutation procedure introduces some kind of randomness which in essence introduces new genes in the solution, this is in order to prevent being trapped into the local optima. This operator is usually used with a small probability, as low as 1%, according to [Article], but the appropriate mutation rate is still an open issue.

Finally the replacement or survivor selection uses the fitness value to identify the individuals which will serve as the parents for the next generations and is responsible to assure the survival of the fittest individuals.

2.6.3 Evolution Strategy

Similar to the Genetic Algorithms, described in the last section, these algorithms try to follow the principles of natural evolution as their method to solve optimization problems. They were introduced during the 0's by Rechenberg [216, 217] and further developed by Schwefel later. The first algorithm used in experimentation was Two membered ES and had a simple mutation-selection scheme.

The basic structure consist on a single parent which produces an offspring following a normally distributed mutation resulting in a $[\lambda]$ number of individuals. Next a selection operator determines the fittest individual which then becomes the parent of the next generation.

The selection process resembles a "extinction of the worst", which means the less fitted individuals are replaced, which sometimes may include the parents, the population is kept with the size $[\lambda]$ constantly in each of the generations.

At the time of its conception the idea of a population wasn't widely used so far, which is why Rechenberg proposed a multi-member ES in [(probably 216 or 217)] where more than one parent can participate in the generation of one offspring individual. This is denoted as $(\mu + 1) - ES$ where μ represents the number of parents.

Two other definitions were then introduced by Schwefel in [237 from the article], which were $(\mu + \lambda) - ES$ and $(\mu, \lambda) - ES$.

- $(\mu + \lambda) - ES$, known as Elitist Evolution Strategy, indicates that μ parents will create $\mu \geq 1$ descendants by recombination and mutation, keeping the best fitted individual of the previous generation and discarding the rest

- $(\mu, \lambda) - ES$, known as Non-Elitist Evolution Strategy, is similar with the difference that none of the parents is kept in any of the subsequent generations, disregarding how good or bad were its fitness compared to the generation they belong.

Two other well-known ES are $(\mu/\rho + \lambda) - ES$ and $(\mu/\rho, \lambda) - ES$, where the ρ refers to the number of parents involved in the procreation the offspring. Mutation in ES can be realised through normally distributed numbers with a mean of 0 and a standard deviation of σ , which represents the the size in the mutation step, when the problem has a continuous solution.

2.6.4 Random search

Also known as Pure Random Search was first defined by Brooks in ... and later named in the classic volumes by Dixon and Szego [44, 45 (from <https://link.springer.com/book/10.1007/978-1-4419-9182-9>)].

It is considered as one of the most basic search strategies. It consists in the following: in each step it creates a new random solution, and the best solution is updated if the new solution outperforms the fit value of the previous best solution.

While this algorithm sacrifices the guarantee of determining the optimal solution within the search space, it can actually be shown that a pure random search converges to the global optimum with a probability of 1.0 according to [<https://link.springer.com/book/10.1007/978-1-4419-9182-9>].

This is most commonly used in addition to other search algorithms compare its performance to random sampling according to [<http://sci-hub.tw/https://doi.org/10.1002/spe.2459>].

2.6.5 Random Descent

Also known as Stochastic Hill Climbing. It is an algorithm similar to regular Hill Climbing, with the difference that it includes a neighborhood which limits the search to only the solutions that are close to the current best solution in each step.

Perhaps the most popular implementation of it was done by Forrest and Mitchell naming it Random Mutation Hill Climbing (RMHC) algorithm (with communication from Richard Palmer) [M. Mitchell and J. H. Holland, "When Will a Genetic Algorithm Outperform Hill Climbing?", in Proceedings of the 5th International Conference on Genetic Algorithms, 1993.]

The local improvements are determined by a neighborhood structure and a fitness function on the Search space of the algorithm. This neighborhood structure can be seen as a undirected graph G on vertex set S . The algorithm checks a member of its neighborhood randomly and transitions to this solution when there is an improvement in the fitness function result or its the same, the process is then repeated in each step.

It was designed with discrete domains in mind, since they have explicit neighborhoods, such as with combinatorial optimization problems. However, even being a stochastic optimization process, it can still get stuck in a local optima.

It is frequently used to have it as a comparison to GA's, according to [<http://papers.nips.cc/paper/1172-stochastic-hillclimbing-as-a-baseline-method-for-evaluating-genetic-algorithms.pdf>]

2.6.6 Parallel Tempering (Replica Exchange Monte Carlo)

It's an algorithm Also known as Replica Exchange Monte Carlo It is considered Similar to Simulated Annealing since it also uses the concept of temperatures, but it has the advantage that takes an advantage by using a multi-core architecture. [<http://www.jamesframework.org/docs/>].

It was originally mentioned as a simulation technique known as replica exchange in a paper by Swendsen and Wang.¹ In this paper, a method of replica Monte Carlo was introduced in which replicas of a system of interest were simulated at a series of different temperatures. Then this replicas undergo a partial exchange of configuration information. The more general form of parallel tempering with complete exchange of configuration information was introduced in 1991 by Geyer according to [doi/10.1039/B509983H].

This algorithm runs several replicas of a particular system of interest, using different temperatures in each one of them. High temperature are able to generate large volumes of phase spaces, while low temperature systems tend to have more precision in the sampling of the phase space and tend to be trapped in a local optima.

This particular implementation of the algorithm uses Metropolis search as the system to replicate, which is an extension of Random Descent where Metropolis where a valid neighbor that is no improvement over the current solution may still be accepted as the new current

solution, and the probability to accept an inferior neighbor is proportional to the difference in the evaluation of the current solution and the temperature of the search.

Using this algorithm requires a minimum and maximum temperature, with each replica of the system with an assigned temperature, after each step the replicas are ordered according to their temperature, and then swapped to push the better solutions to the lowest temperatures, so they can eventually converge while higher temperature replicas are constantly modified in a search for improvement. Each of the replicas apply repeatedly a series of moves to a private solution, and then the global solution is tracked in the main search.

2.7 Multi-Objective Algorithms

2.7.1 ESPEA

Stands for Electrostatic Potential Energy Evolutionary Algorithm and its a very recent algorithm that generates several Pareto front approximations, focusing only in the solutions that appear interesting to the decision maker. It was first proposed in [<https://sci-hub.se/https://dl.acm.org/citation.cfm?id=2754674>] by [Authors]. It was proposed because it was noticed that many of the algorithms found in the literature make use of popular crowding distance metrics that doesn't necessarily converge into optimality according to [13 from <https://sci-hub.se/https://dl.acm.org/citation.cfm?id=2754674>]. It also has been shown in [28 from <https://sci-hub.se/https://dl.acm.org/>] that an optimal distribution depends on the chosen reference point, but these methods are not adequate for constrained problems or discontinuous Pareto fronts.

The algorithm attempts to design a electromagnetism inspired heuristic, each solution has an assigned charge based on how close it is to a randomly selected subset from an archive of non-dominated solution. Then the charges are translated to force vectors which move the solutions in the search space. The best solutions are stored in an archive that uses clustering and crowding distance to prune the solutions.

2.7.2 MOMBI2

MOMBI stands for Many Objective Metaheuristic Based on the R2 indicator introduced first in [<https://sci-hub.se/10.1145/2739480.2754776>] by Carlos A. Coello. It has an advantage over other many-objective optimization algorithms found in the literature since a lot of them such as NSGA-II relies on the Hypervolume indicator which can become a costly function as the number of objectives grow, MOMBI2 uses the R2 indicator instead [8 from the article].

The R2 indicator is considered part of the R indicator family first introduced by Hansen and Jaszkiewicz in 1998 [5 from <http://www.cmapx.polytechnique.fr/dimo.brockhoff/publicationListFiles>]. All of them are based on a set of utility functions, in this case comparing 2 sets of points being a reference pareto front and another front, having several variants introduced at the time, with the variation being the way the utility functions are evaluated and combined: - R1, The ratio of the set being better than the other - R2, The mean difference in utilities - R3, The mean relative difference of in utilities

In particular the R2 definition has been broadly used and is one of the most recommended performance indicators [8 from the article] along with Hypervolume and it is highly correlated to it [<https://sci-hub.se/10.1145/2739480.2754776>].

R2 is used because it induces the complete ranking in the set of all approximations. It is based on the assumption that it is allowed to add values of different utility functions in the set. Therefore it is also dependent on the scaling of its utility functions [<https://sci-hub.se/10.1145/2739480.2754776>].

The first version of MOMBI was introduced in [9 of <https://sci-hub.se/10.1145/2739480.2754776>] but it had the problem that the it presented a loss of diversity in the solutions as the dimensions of the problem grew to address this problem the Chevishev function CHE set was changed to the Achievement Scalarizing Function ASF which is similar to the former but introduces some normalization.

The algorithm works as follows, each iteration new individuals are created from the parents selected randomly here the looking for those individuals that minimized ASF

The maximum values of each objective function constitute a point z_{max} , it is ensured that there are not repeated individuals and the size of the population is at least the desired size, if these requirements are not satisfied, then the z_{max} point is penalized by updating it with the worse values of the whole population.

In the following steps, the population is ranked and then reduced to the desired size.

2.7.3 NSGAII

NSGA-II

Is one of the most known Multi-Objective optimization methods, it was first proposed by Shinivas and Deb in 1994 for its first version, and then later in 2002 as NSGA-II following the work from Shinivas and other authors, this algorithm heavily relies on the concept of dominance introduced in section [ref]. In addition it also makes use of the Crowding Distance which is a certain preference for solutions that are less crowded in the solution space.

NSGAII was introduced as a response after the criticism of the first version NSGAII, which had the following problems, each one which has been addressed in this new version:

A high computational complexity of the non-dominated sorting, which had a complexity of $O(MN^3)$, where M is the number of objectives and N is the size of the population, this new version has instead a complexity of $O(M(2N)^2)$, the algorithm for this is shown in [fig]. Another problem was the lack of elitism, since [25], [18] have shown that elitism can speed up the performance of the GA significantly, the elitism as described in the next paragraph is now assure by keeping all the solutions from the previous generation when the ranking is done. Finally a need for specifying a sharing parameter, since traditional mechanisms use the concept of sharing to keep a wide variety of equivalent solutions.

This algorithms works as follows:

After an initial population is created, each individual is ranked and sorted according to their dominance, each solution then is assigned a fitness according to their non-domination level (1 is assigned to the best, then 2 and so on). This for minimization problems. Then a binary tournament selection, crossover and mutation take place and an offspring population of size N is created, the process is then repeated joining the previous generation and their

offspring as the same population, and it continues with the population always taking the parents of the previous generation, therefore Elitism is ensured. Is important to notice that in the following generations, the ranking will also consider the crowding distance for the ranking.

2.7.4 RandomSearch

2.7.5 SPEA2

2.8 Comparison Between Single-Objective and Multi-Objective Optimization

Existen distintos trabajos que contemplan una comparación entre estrategias mono-objetivo y multiobjetivo para distintos campos de la industria, por ejemplo Jiao, Zeng[?] entre otros hacen uso de distintas estrategias para convertir un problema mono-objetivo en uno multi-objetivo dinámico en el cual aplican un algoritmo evolutivo multiobjetivo con una mejora en los resultados comparándola con las soluciones existentes. Otro tema muy recurrente en los cuales se comparan su desempeño es en el diseño de antenas para mejorar la recepción telefónica, como es el caso del trabajo de Rahmat-Samil y Nanbo Jin.[?]

Por último Battiti y Passerini[?] hablan sobre el uso de algoritmos genéticos que se adaptan al tomador de decisiones, esto resulta relevante porque en la plataforma final el usuario propiamente no tiene una decisión real sobre a qué grupo terminará siendo asignado sino que esto se pretende realizar de manera automática, Battiti menciona algunas de las estrategias posibles para que pueda adaptarse el comportamiento de un tomador de decisiones y además discute algunas de las limitaciones que un sistema de este tipo presentaría de acuerdo con sus preferencias.[?]

2.9 Metrics

2.9.1 Epsilon

2.9.2 Spread

2.9.3 Generational Distance

2.9.4 Inverted Generational Distance

2.9.5 Inverted Generational Distance +

2.9.6 Hypervolume

2.9.7 Friedman Test

2.10 Frameworks

2.10.1 jMetal

jMetal es un framework basado en el lenguaje de programación Java, orientado a objetos enfocado al desarrollo, la experimentación y el estudio de metaheurísticas para resolver problemas de optimización multiobjetivo. jMetal incluye diversos optimizadores de los más comunes, además de una serie de problemas además de algunos de los indicadores más conocidos para medir el desempeño de los algoritmos. Incluye herramientas para llevar a cabo estudios experimentales que pueden ser configurados desde su interfaz para generar reportes de forma estadística de los resultados obtenidos. También es posible aprovechar el uso de procesadores multi-core para hacer el proceso de experimentación más rápido.

Gran parte de los problemas de optimización en el mundo real son multiobjetivos, lo que significa que resolverlos requiere la optimización de 2 o más funciones u objetivos contradictorios. Este tipo de problemas se le conoce como Problemas de optimización multiobjetivo (MOP's).

Para resolver este tipo de problemas usualmente no es útil el uso de técnicas exactas, por lo que suelen usarse métodos de aproximación. Tal y como en las optimizaciones multiobjetivo, se hace uso de metaheurísticas para resolver estos MOP's. Los algoritmos evolutivos resultan ser muy populares en este aspecto, algunos de estos algoritmos caen en esta categoría, como son el caso de NSGA-II, PAES y SPEA2.

Para poder medir de forma correcta el funcionamiento de cada uno de estos algoritmos además de compararlos entre sí, para poder elegir de manera oportuna el que tenga el mejor funcionamiento se hace uso de software especializado que recoja datos experimentales para cada una de las pruebas que se realizan, además de tener el mismo tipo de problema a resolver para tener un punto de comparación más adecuado.[12]

Algunas de las características que se busca formen parte de este tipo de software son:

- Incluir los algoritmos más actuales disponibles
- Contener las pruebas de desempeño más aceptadas para la resolución de problemas multiobjetivo
- Dar indicadores de calidad para medir el desempeño de cada una de las pruebas y asistir de manera más accesible en las investigaciones de sus usuarios.

jMetal por su parte cuenta con las siguientes características que hacen la plataforma una herramienta única comparada con las alternativas existentes:

- La validación de la implementación, se comparó las implementaciones incluidas en jMetal de los algoritmos NSGA-II y SPA2 con sus versiones originales obteniendo resultados competitivos.

2.10.2 James

James is a framework for discrete optimization, that uses a variety of local search metaheuristics. Problems can be defined by providing a configuration for the problem and the solution. This framework was added after realizing that jMetal lacked many significant Single-Objective Algorithms to be used for comparison.

However, to make compatible between each other, all the problems were defined as jMetal problems and an adapter was used for conversion between both frameworks.

2.11 Psychometrics?

2.12 AMI Meeting Corpus

Ya que no hay ningún paper que diga que función usar para esto se le hizo un análisis de datos al AMI meeting corpus para ver qué relación había entre el número de silencios y la homogeniedad de las participaciones de acuerdo con los functional roles de Benne y Sheats, y pues resulta que si hay una correlación con respecto al porcentaje de participaciones por lo que se usó esta medida para tratar de predecir la interacción del grupo, considerando que esta participación debe ser homogénea entre ellos por lo que es el inverso a la suma de las participaciones. [...] Esta ha sido usada en varios trabajos como para predecir estilos de participación como es el caso de Vinciarelli[?] quien usa esta base de datos para identificar los roles funcionales de Benne y Sheats[?].

2.13 Roles funcionales de Benne and Sheats

Los roles funcionales de Benne y Sheats son una herramienta que ayuda a definir que tipo de reacción tiene una persona al momento de entablar una conversación[ref]. Los roles en

los que estamos interesados específicamente se conocen como los **Roles de mantenimiento** o sociales que están basados en el trabajo de Biddle [ref] sobre la teoría de los roles, ya que el resto de las definiciones de roles que son los enfocados a tareas y disfuncionales no proporcionan información relevante para la investigación. Estos roles en específico se definen a continuación:

- **Atacante:** Es aquella persona que invalida los comentarios de otros y "ataca" al resto de los miembros del grupo.
- **Protagonista:** es aquella persona que inicia las conversaciones, asume un rol de autoridad. En este caso es un rol muy valioso para nuestra investigación.
- **Seguidor (Supporter):** esta persona presenta una actitud cooperativa, pone atención a la plática, acepta las propuestas y ofrece soporte técnico.
- **Gatekeeper:** es una persona que desempeña el rol de moderador, revisando que cada persona tenga su oportunidad de participar de forma similar.
- **Neutral:** Es similar al rol de seguidor, sólo que en lugar de participar de forma activa, acepta los comentarios de forma pasiva.

Cabe destacar que una persona puede representar distintos tipos de roles en una misma conversación, aunque generalmente se le clasifica dependiendo de cual fue su rol más prominente, lo cual será usado como ventaja, ya que en la base de datos de **AMI meeting corpus** las etiquetas de los roles están dadas para cada participación y no necesariamente para cada persona, por lo que nos es posible determinar que porcentaje represento cada rol del tiempo total de lo que duro la conversación.

2.14 Facebook Ad Platform

La red social conocida como Facebook es útil hoy en día para mantener el contacto entre amigos y conocidos, sin embargo también es una útil herramienta de mercadotecnia para dar a los usuarios que pertenecen a un cierto mercado anuncios enfocados a cada uno de ellos acorde a sus preferencias, comportamientos y ubicación, esto de hecho sigue una metodología de la mercadotecnia conocida como Demografía, comportamientos e intereses[ref]. Por lo que para esta investigación respecta solo nos interesa la cuestión de los intereses.

Estos intereses están determinados por la clasificación de las paginas o grupos a las que el usuario sigue, sin embargo cuenta con una clasificación más general a manera de una ontología que Facebook ha estado construyendo a lo largo de los años usando distintas técnicas de clustering para que pueda tener una categorización más general de los distintos tipos de intereses de cada uno de los usuarios.

Además Facebook permite que a través de una plataforma conocida como Facebook Ad platform en la cual los negocios pueden hacer uso de ella para publicar distintos tipos de anuncios dirigidos a un tipo de mercado de acuerdo con los intereses o comportamientos que cada uno describe, e. esta plataforma es publica y puede hacerse uso de ella para determinar un costo aproximado al momento de preparar campañas de mercadotecnia. Sin embargo para esta investigación se usa con otro tipo de propósito, en este caso se usa para determinar el tamaño de población para cada uno de los intereses que se describe como será mencionado en las secciones [ref1] y [ref2].

También esta plataforma puede ser usada al mismo tiempo para generar la distribución de los datos ya que esta directamente ligada con la ontología y puede ser consultada para medir las distancias entre un interés y otro. Parte de la ontología puede ser vista en la Fig [ref]

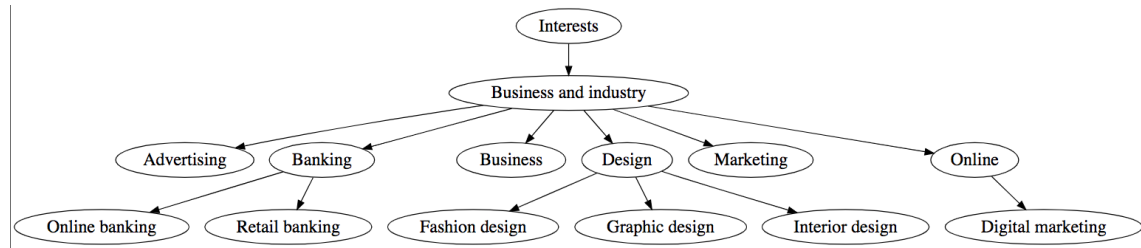


Figure 2.1: Caption

La ontología que ha creado Facebook se determina con un árbol de 8 intereses generales y 314 específicos cada uno clasificado según la relación a un interés general, esto nos permite buscar intereses específicos en común, por ejemplo si una persona le gustan los programas de televisión podemos asumir también que le gustan los reality shows o los programas de concursos etc, de esta forma otra persona que tenga un gusto de por ejemplo de series de televisión, tiene una relación directa con la primera y serán capaces de encontrar algún tema en común para iniciar una conversación.

2.15 CEFR

Conocida como **Common European Framework of Reference for Languages** por sus siglas en Inglés, es una herramienta usada a nivel internacional para evaluar el nivel de conocimiento de un idioma extranjero y es comúnmente conocido a lo largo del mundo de tal manera que aunque se haya realizado un test diferente, es posible encontrar un tipo de equivalencia usando esta clasificación, lo que nos permite ser capaces de tomar en cuenta en nivel de una persona sin importar que tipo de test tomo como información al registrarse en esta plataforma, o inclusive usando la descripción de cada uno de los niveles definirse a si mismo como perteneciente a alguno de ellos. A continuación se describen los niveles y el significado de cada uno:

Chapter 3

Methodology

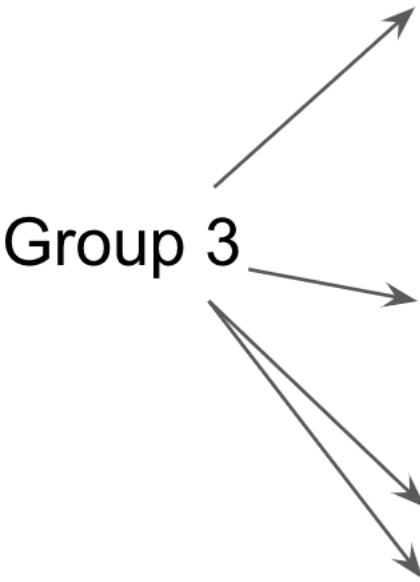
In the following chapter is explained the different parts of the problem to solve and

3.1 Problem Design

The problem established is described as given a database of n users, an optimal arrangement of these users must be found, this arrangements puts each user in a group, with the users being in the same group that optimizes their predefined preference as described in [ref I], this optimization is described as finding the less distance between the level of experience of the users inside the same group, finding the less distance among an ontology of interests, where similar interests related interests are in the same branch, as well as the having a group size near the optimal group size which is defined as between 4 and 5, and the percentage of participation which is how much the users may participate in the group, maximizing in the most participation.

The solution is then the number of the group for each user, serving it as an id only, without any relevance in the order the groups are presented, only that the preference of the users is maintained within the group, this can more clearly be seen in ref[fig.] and [fig.]

Figure 3.1: This is an example of a dataset of 9 users, each with their id, exp, 3 different interests and a participation percentage



id	Exp	Interest #1	Interest #2
0	4	Photography	Vehicles
1	3	Television programme	TV chat shows
2	6	Music	Television programme
3	5	Films	Family
4	3	Music	Television programme
5	4	Games	Television programme
6	4	Television programme	Restaurants
7	4	Television programme	TV chat shows
8	5	Television programme	TV chat shows
9	4	Restaurants	Coffeehouses

3.2 Objective Functions

En la siguiente sección se describen distintas funciones objetivo para predecir la preferencia del usuario y la estabilidad del grupo, buscando minimizar el valor de cada una de ellas.

3.2.1 Tamaño del Grupo

Para propósitos de la investigación el tamaño ideal del grupo se considera como de **4 a 5** personas, ya que este tamaño usualmente permite una participación de todos los usuarios que pertenezcan al grupo. Es posible que puedan existir grupos de **3 o 6** personas, pero esto no es lo ideal, sin embargo se deja abierta la posibilidad en caso de que alguna otra característica como los intereses en común o el estilo de participación resulten más relevantes para el grupo que su tamaño.

La función para calcular este objetivo mide la cercanía al valor **4.5**, ya que 4 y 5 se consideran igual de ideales y 4.5 es simplemente el promedio entre ambos valores. Esta función está definida a continuación:??

$$f = |groupSize - 4.5| \quad (3.1)$$

User 2	Interests	Family	Dating	Fitness	Running	Pets	Hobbies	Dogs
Dating	2	1	0					
Running	2			1	0			
Dogs	3					1	2	0
Dating	0.3333333	0.5	1					
Running	0.3333333			0.5	1			
Dogs	0.25					0.5	0.3333333333	1

Figure 3.2: Tabla de valores de Intereses

	Interests	Family	Dating	Fitness	Running	Pets	Hobbies	Dogs
User 2	0.3333333333	0.5	1	0.5	1	0.5	0.3333333333	1

Figure 3.3: Vector de Intereses

3.2.2 Nivel de experiencia

El nivel de experiencia en el lenguaje hace referencia al estándar CEFR[1]. Los valores que se consideran para ello son A1, A2, B1, B2, C1 y C2, por lo que se les asigna un valor numérico a cada uno siendo A1 0 y C2 5. Lo que se busca principalmente es que los usuarios dentro del grupo no tengan mucha variación en cuanto a su nivel de experiencia. De esta forma el vocabulario usado no se considera tan avanzado para integrantes con un nivel bajo ni tan sencillo para integrantes más avanzados. La fórmula ?? consiste en una desviación estándar del valor del nivel de cada uno de los integrantes, de forma ideal que sea alcanzado el 0 indicando que todos los usuarios del grupo forman parte del mismo.

$$\sigma = \sqrt{\frac{1}{n-1} \sum_{i=1}^n (x_i - \bar{x})^2} \quad (3.2)$$

3.2.3 Intereses

Para calcular la similitud entre intereses se usa una estrategia similar a Madylova y Gunduz [2] donde los intereses son considerados como un vector valores en los que se señala una cantidad que indica el orden jerárquico de acuerdo a la ontología definida por la red social de Facebook, por ejemplo si un usuario tiene como intereses: Los perros, Correr y las Citas, su vector de intereses se estaría dado por los valores en la Figura ?? . En la parte de arriba se indica el valor de la jerarquía, en la parte de abajo se normaliza usando $1/(1+v)$, después se eligen los valores mayores de cada una de las columnas de esta forma da el vector resultante como se muestra en la Figura ??

Para calcular que tan similar es un usuario con respecto a otro se usa la formula de distancia cosenoidal, que se muestra en la ecuación ?? donde A es el vector de intereses del primer usuario y B es el vector de intereses del segundo. Cabe destacar que es muy frecuente que los vectores de intereses de los usuarios contengan ceros al momento de compararlos, de tal manera que si el resultado de la función da como resultado exactamente 0, significa que ambos usuarios no tienen nada en común.

$$\cos(\theta) = \frac{\sum_{i=1}^n A_i B_i}{\sqrt{\sum_{i=1}^n A_i^2} \sqrt{\sum_{i=1}^n B_i^2}} \quad (3.3)$$

$$f = 1 - \cos(\theta) \quad (3.4)$$

3.2.4 Estilo de Participación

Para esta función trato de buscarse en la literatura algún tipo de función matemática ya definida para evaluar que tipo de roles son los que provocan que haya una participación más homogénea dentro del grupo dependiendo del rol funcional de cada uno de los participantes y su compatibilidad entre si. Sin embargo no se encontró ninguna, por lo que fue necesario definirla por medios propios. Afortunadamente se cuenta con una definición concreta de como debe ser la función, tomar en cuenta como parámetro los roles funcionales de cada usuario y dando como resultado de su evaluación el nivel de participación del grupo de forma numérica con la idea de optimizar dicho resultado.

Es por esto que al hacer el análisis de los datos como se describe en la sección de Experimentación [ref], se toman en cuenta

3.3 Mutación

Una mutación de acuerdo a los algoritmos genéticos debe de introducir al sistema una solución que no se exista aún. Sin embargo no es suficiente con cambiar un usuario por otro en un grupo para la mutación ya que este usuario probablemente se encuentre en otro grupo diferente y el sistema prohíbe que un usuario este en 2 grupos distintos a la vez. Por ello el algoritmo para la mutación toma un grupo de origen al azar, quita un usuario de un grupo y lo pone en otro, tomando en cuenta las restricciones del tamaño del grupo, es decir, si es un grupo que quitándole un usuario resultaría menor a 3 entonces se debe seleccionar otro grupo. De forma similar para el grupo destino se verifica que no sobrepase el tamaño máximo de 6 una vez que el nuevo usuario sea agregado, y si es el caso se selecciona otro grupo. El algoritmo completo se presenta como pseudocódigo a continuación:

??.

jMetal cuenta con distintos algoritmos para realizar la mutación, sin embargo estos comprenden únicamente problemas binarios o de permutación, y ya que es necesario un algoritmo para obtener una solución combinatoria, fue necesaria la creación de un nuevo algoritmo que a continuación se propone.

Resulta trivial hacer esta mutación sin verificar que se repita el usuario, ya que el tiempo requerido para reparar la solución es el mismo que revisarla de forma previa, pero no se descarta que sea posible hacer obtener soluciones sin restricciones y repararlas al final.

Algorithm 1 Group Combination Mutation

```

1: procedure MUTATION( $s$ )
2:    $g1 \leftarrow getRandomVariable(s)$   $\triangleright$  Obtiene una variable aleatoria de la solución
3:    $g2 \leftarrow getRandomVariable(s)$ 
4:   while  $groupSize(g1) \leq minSize() + 1$  do  $\triangleright$  Revisa si en grupo de origen es mayor
      que el tamaño menor posible
5:      $g1 \leftarrow getRandomVariable(s)$ 
6:   while  $groupSize(g2) > maxSize() - 1$  do  $\triangleright$  Revisa si en grupo destino es menor
      que el tamaño mayor posible
7:      $g2 \leftarrow getRandomVariable(s)$ 
8:    $u \leftarrow getRandomUserFromGroup(g1)$ 
9:    $addUserToGroup(u, g2)$ 
10:   $removeUserFromGroup(u, g1)$ 

```

3.4 Crossover

La herramienta de jMetal cuenta con distintos algoritmos de crossover, y ya que estos únicamente se basan en los cromosomas que corresponden a los grupos creados, es posible usar cualquier algoritmo de esta herramienta para generar soluciones nuevas en la población. El algoritmo que se usó en cuestión recibe el nombre de N-Point Crossover, donde la N corresponde al número de cromosomas que se cruzaran, es decir suponiendo que $n = 10$ entonces 10 grupos se tomarán de una solución y se pasarán a otra que a su vez también intercambiará 10 grupos y el resto... ya que se busca que el crossover se realice justo a la mitad, esta N corresponde al número de cromosomas/variables o bien el número de grupos.

Para la primera fase los experimentos que se definieron consistieron en probar únicamente las funciones objetivo de forma individual y determinar su factibilidad con respecto a si son reproducibles y pueden ser útiles para cualquier tipo de solución generada.

Chapter 4

Experimentos Preliminares

4.1 Creación de la Base de Datos

Debido a que esta investigación se lleva a cabo con un concepto nuevo, no existe ninguna base de datos que tenga todos los datos relacionados al respecto, por lo que se optó por crear una base de datos artificial alimentada con la distribución de distintas estadísticas que se encontraron en diversos estudios, para poder acercarnos a lo que podría ser la distribución que se pudiera obtener cuando la plataforma este lista.

4.1.1 Datos sintéticos

Se hizo un estudio de como pueden ser usados los datos sintéticos en [ref1] y [ref2], además de esto los datos sintéticos pueden servir para evitar problemas de privacidad para los usuarios cuyos datos se encuentran en la base de datos.

4.1.2 Datos de Idiomas

Para los datos de idiomas se busco en distintas fuentes algún tipo de estadística que pudiera definir como se comporta la distribución entre los distintos niveles de idiomas que forman parte del CEFR, de esta manera hacer una distribución similar para nuestra base de datos artificial.

4.1.3 Datos de Intereses de Facebook

Como previamente se mencionó Facebook cuenta con una base de datos de intereses que son distribuidos entre varias cosas por intereses ligados a una población, para poder calcular la distribución se calculó un total de usuarios de lo que contempla México, después se ajustó para que arrojará resultados por cada interés, en cada caso la herramienta daba un número que consistía en un aproximado de la segmentación de ese interés en particular. Cabe destacar que hay intereses generales que conforman la parte superior de la ontología, directamente después del origen, que no fueron tomados en cuenta porque se consideraron muy generales y resultaban llevarse un [prc]% de la distribución total, por lo que únicamente se tomaron en cuenta intereses de tercer y cuarto nivel, que se acercan más a los intereses específicos que

podría presentar cada usuario. Por último, también es importante señalar que no se trabaja con números exactos con esta herramienta, sin embargo para propósitos de esta investigación sólo es importante obtener un porcentaje aproximado, por lo que este detalle resulta no muy relevante.

4.2 Función de Distancia entre intereses

Para este experimento se generaron 4 usuarios de forma aleatoria con sus correspondientes vectores de intereses, del mismo modo que se definieron en la Metodología ???. La idea era buscar la forma de encontrar la distancia entre 2 vectores o un grupo de vectores de ser posible, con un valor que idealmente fuera de 0 a 1, de forma continua. Para ello se compararon distintas medidas de similaridad, algunas de ellas encontradas en [?], [?], un factor importante para usar como punto de comparación es que esta función es que permitiera trabajar con valores de 0, lo cual terminó descartando gran parte de las funciones, ya que su resultado hubiera sido dividido entre 0. Al final se hizo una comparativa entre 2 medidas, la distancia cosenoidal y el coeficiente de correlación de Pearson,

4.3 Función de Participación

Al momento de buscar alguna métrica que ayudara a definir la estabilidad del grupo entre los distintos estilos de participación, no fue posible encontrar en la literatura algún valor con suficiente validez para usarse de la forma que requiere esta investigación. Por lo que fue necesario definirla de otra manera, para ello se hizo un análisis de datos a la base de datos pública de conversaciones conocida como AMI Meeting Corpus buscando alguna correlación entre los roles funcionales y la homogeneidad de participación, que en este caso se le considera como el menor número de silencios y tiempos de participaciones aproximadamente similares.

Se hizo un análisis de distintas conversaciones que tuvieran los roles funcionales indicados, afortunadamente esta base de datos cuenta con información de al menos 8 conversaciones independientes, con 4 grupos de personas distintos, teniendo 2 conversaciones por grupo.

De forma preliminar como se puede observar en la Fig ?? que existe la posibilidad de una correlación fuerte entre el número de silencios con la cantidad total de "protagonismos", cabe mencionar que en este caso los roles no son características de la persona, si no de la participación, así cada vez que una persona participa puede hacerlo llenando un rol distinto, el rol que nos interesa en este caso es el rol de "protagonista" ya que según su descripción es aquella persona que normalmente inicia las conversaciones.

4.4 Optimización mono-objetivo

Para la optimización mono-objetivo se definió una función compuesta que puede apreciarse en la función ??, por las funciones objetivo previamente definidas considerando sus pesos de forma uniforme en esta parte de la investigación. En el resto de los experimentos, esta función también se utiliza como principal punto de comparación entre las distintas estrategias mono-objetivo y multi-objetivo.

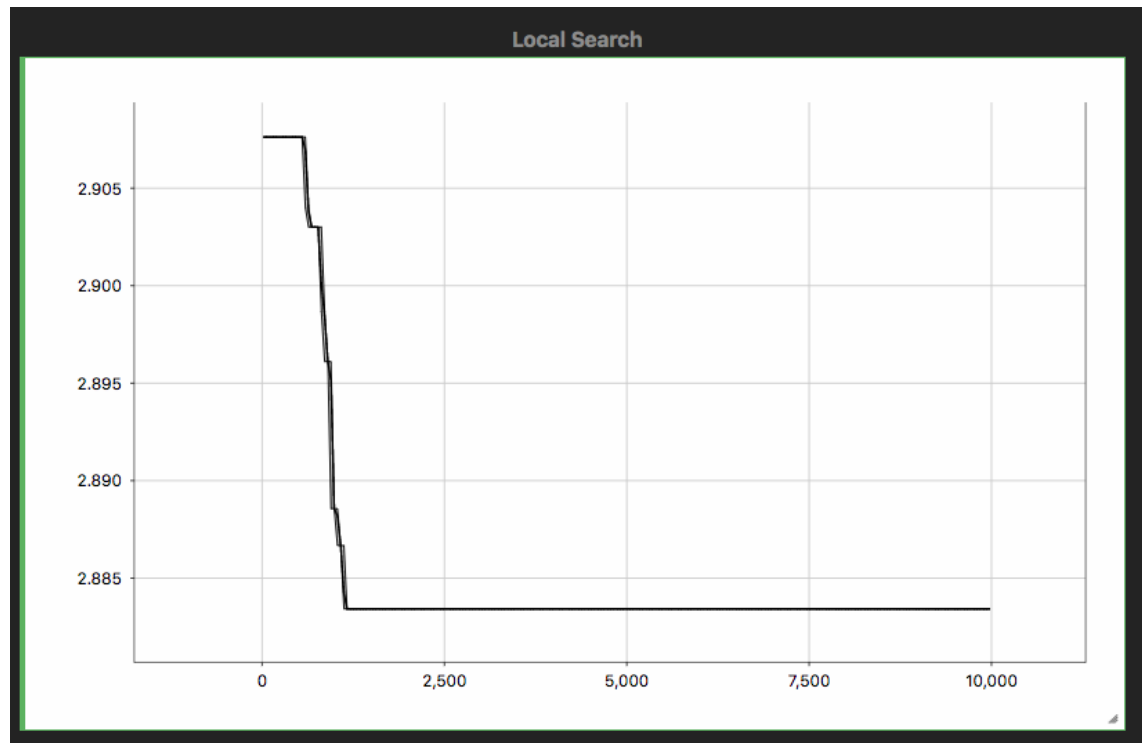


Figure 4.1: Búsqueda Local

- La función del tamaño del grupo f_1
- La función de la similitud de intereses f_2
- La función de la similitud del nivel f_3
- La función del estilo de participación f_4

$$f = w_1 \times f_1 + w_2 \times f_2 + w_3 \times f_3 + w_4 \times f_4 \quad (4.1)$$

4.5 Búsqueda local

Para comenzar a experimentar se usó el algoritmo de búsqueda local, ya que este es considerado una de las formas más directas para realizar la optimización, en caso de que las soluciones que produjera se encontraran dentro del frente de pareto no sería necesario usar algoritmos más avanzados. Como era de esperarse, este algoritmo se queda trabado en un mínimo local, inclusive después de un número significativo de iteraciones no logra mejorar mucho, por lo que queda descartada la idea de que encuentre una solución dentro del frente de pareto

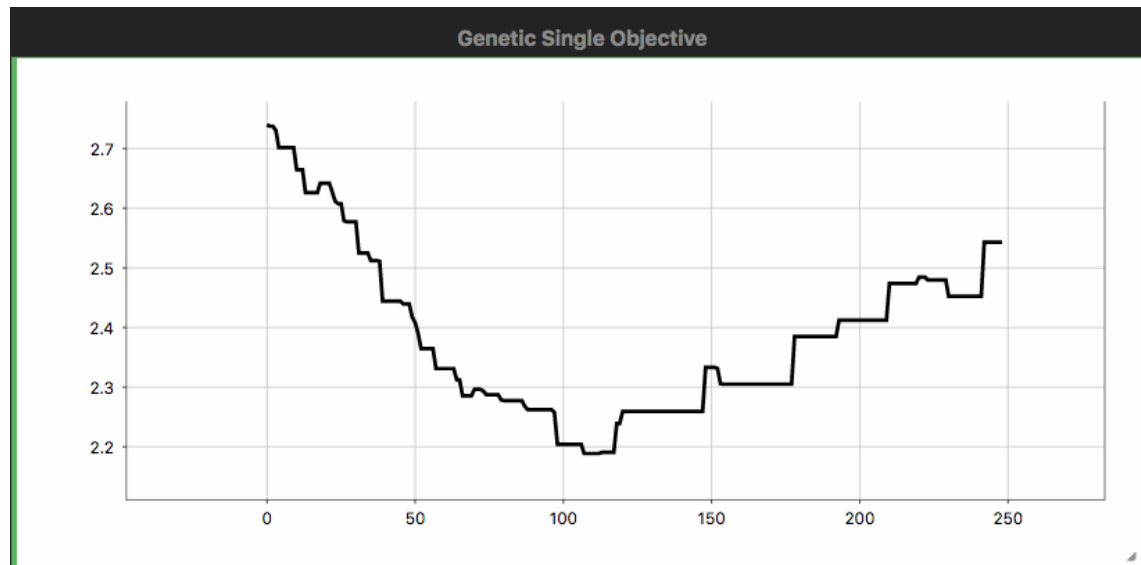


Figure 4.2: Algoritmo Genético Mono-objetivo

4.6 Algoritmo Genético

Debido a que la optimización Multi-objetivo se encuentra dominada en la literatura por algoritmos genéticos, resulta relevante usar un algoritmo genético mono-objetivo para hacer una comparación más directa, ya que ambos comparten parámetros como los algoritmos de mutación y crossover además de un ϵ para la evaluación de las soluciones. Como podemos ver en los resultados estas soluciones resultan ser mejores que las obtenidas por búsqueda local y si observamos la muestra de una de las soluciones es evidente ver que la estabilidad del grupo resulta mejor comparada con la que resultaría de un grupo generado de forma aleatoria.

4.7 Optimización Multi-Objetivo

Para la optimización multiobjetivo se utilizó el algoritmo NSGA-II qué significa... ya que es uno de los más usados en la literatura de hoy en día, además de que no necesitan parámetros adicionales a los usados para el algoritmo genético de optimización mono-objetivo.

Es posible ver en estos resultados que las soluciones obtenidas por la optimización multiobjetivo resultan ser significativamente mejores, ya que al sumar los valores de las soluciones resultan ser menores a las obtenidas por una optimización mono-objetivo por algoritmos genéticos.

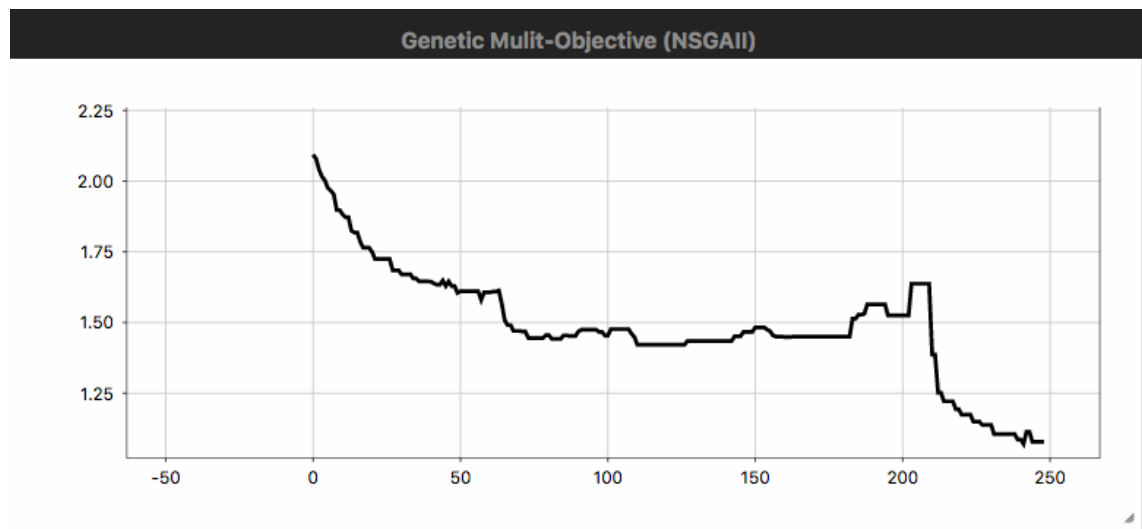


Figure 4.3: Multi-Objetivo (NSGAI)

Chapter 5

Experiments

For the experimentation phase, all the parameters were left as their default values indicated by the jMetal and James Framework, while they could be adjusted manually to improve their individual performance, this is out of the scope for this research. the only parameters adjusted to improve were the population size and number of generations (number of iterations for non-genetic algorithms).

As indicated by [<https://link.springer.com/book/10.1007/978-1-4419-9182-9>] the performance is sensitive to the value of the population in the case of the genetic algorithms, if the population is too large, then it is essentially pure random search, so in essence, the parameters of an algorithm outperforming random search should be considered as an ideal size for the population.

5.0.1 Algorithm Implementations

Here the algorithms finally used are described in their specific implementation for the problem, as previously noted, all the implementations were already included in the jMetal and JAMES frameworks, and the only definitions created were the mutation and crossover procedures and the problem and solution definitions.

5.0.2 Single-Objective Algorithms

Local Search

In the framework used jMetal, didn't include the implementation as the for the other algorithms, but it did have as middle step for other algorithms as ABYSS, but here is used as a standalone algorithm. With this in mind, for the perturbation step the procedure of mutation is used, as it complies with the requirements of not being too weak in order to escape the basin of attraction or too strong to make it too similar o multistarch local search. These criteria defined in [doi.org/10.1016/j.ins.2013.02.041].

5.0.3 Genetic Generational Algorithmn

Is an implementation of the Genetic Algorithm which. . .

5.0.4 Genetic Steady Algorithmn

Is another implementation of the Genetic Algorithm which. . .

5.0.5 Elitist

Is an implementation of the Evolution Strategy which. . .

In this case the parent is preserved each generation, which in essence means that if there is no offspring considered better, the parent then will produce another set of offspring, until a better individual is produced.

In the case of Evolutionary Strategies Elitist and Non-Elitist, the mutation should follow a normal distribution, but since the solutions are discrete the solutions are generated randomly using the mutation operator as with all the other algorithms.

5.0.6 Non-Elitist

Is another implementation of the Evolution Strategy which. . .

5.0.7 Random search

It should be noted that Random Search is used for comparison, because it doesn't perform any improvement in a given solution, it would be like generating a random set of solutions and selecting the best a determined number of times.

5.0.8 Random Descent (Replica Exchange Monte Carlo)**5.0.9 Tabu Search****5.0.10 Parallel Tempering****5.1 Multi-Objective Algorithms****5.2 Reference Front****5.3 Parameters Definition****5.4 Comparison between Single-Objective Algorithms****5.4.1 Epsilon****5.4.2 Spread****5.4.3 Generational Distance****5.4.4 Inverted Generational Distance****5.4.5 Inverted Generational Distance +****5.4.6 Hyper-volume****5.4.7 Friedman Test****5.5 Comparison between Multi-Objective Algorithms****5.6 Comparison between Single-Objective and Multi-Objective Algorithms****5.7 Hyper-parameters****5.8 Unfair comparison?****5.9 Building pareto front from Single-Objective Algorithms**

The First results (Recover data from the presentation), showed that apparently the random algorithms surpassed overall the other, this is because (note of the beginning)

Appendix A

Appendix

Your appendix should go here.

Curriculum Vitae

(Only one page) Master student was born in Monterrey, México, on December 25, 2208. He earned the Systems Engineering degree from the Instituto Tecnológico y de Estudios Superiores de Monterrey, Monterrey Campus in May 2008. He was accepted in the graduate programs in Information Technologies and Electronics in May 2010.

This document was typed in using L^AT_EX 2_ε^a by Miguel Angel Bravo Vidales (Student).

^aThe style file `phdThesisFormat.sty` used to set up this thesis was prepared by the Center of Intelligent Systems of the Instituto Tecnológico y de Estudios Superiores de Monterrey, Monterrey Campus