HIBRIDACIÓN DE LA MEJOR BÚSQUEDA ARMÓNICA GLOBAL Y EL ALGORITMO K-MEANS PARA EL CLUSTERING DE DOCUMENTOS WEB

ANEXOS

JENNIFER KATTERINE ANDRADE ROJAS
WILLIAM ANDRÉS CONSTAÍN DÍAZ

Director: Mag. CARLOS ALBERTO COBOS LOZADA

UNIVERSIDAD DEL CAUCA
FACULTAD DE INGENIERÍA ELECTRÓNICA Y TELECOMUNICACIONES
DEPARTAMENTO DE SISTEMAS
POPAYÁN, FEBRERO DE 2010

TABLA DE CONTENIDO

LISTA DE TABLAS

LISTA DE FIGURAS

ANEXO A – LISTA DE STOPWORDS Y VERBOS EN INGLÉS

1. LISTA DE STOPWORS

A continuación se presenta la lista de Stop Words que se utilizó en la fase de pre-procesamiento del algoritmo IGBHSK, la cual se tomó del sitio de la Universidad UNINE, de Chile, específicamente de la página web "IR Multilingual Resources at UniNE" (http://www.unine.ch/jacques.savoy/clef/index.html).

English StopWords (palabras en Inglés).

"a", "a's", "able", "about", "above", "according", "accordingly", "across", "actually", "after", "afterwards", "again", "against", "ain't", "all", "allow", "allows", "almost", "alone", "along", "already", "also", "although", "always", "am", "among", "amongst", "an", "and", "another", "any", "anybody", "anyhow", "anyone", "anything", "anyway", "anyways", "anywhere", "apart", "appear", "appreciate", "appropriate", "are", "aren't", "around", "as", "aside", "ask", "asking", "associated", "at", "available", "away", "awfully", "b", "be", "became", "because", "become", "becomes", "becoming", "been", "before", "beforehand", "behind", "being", "believe", "below", "beside", "besides", "best", "better", "between", "beyond", "both", "brief", "but", "by", "c", "c'mon", "c's", "came", "can", "can't", "cannot", "cant", "cause", "causes", "certain", "certainly", "changes", "clearly", "co", "com", "come", "comes", "concerning", "consequently", "consider", "considering", "contain", "containing", "contains", "corresponding", "could", "couldn't", "course", "currently", "d", "definitely", "described", "despite", "did", "didn't", "different", "do", "does", "doesn't", "doing", "don't", "done", "down", "downwards", "during", "e", "each", "edu", "eg", "eight", "either", "else", "elsewhere", "enough", "entirely", "especially", "et", "etc", "even", "ever", "every", "everybody", "everyone", "everything", "everywhere", "ex", "exactly", "example", "except", "f", "far", "few", "fifth", "first", "five", "followed", "following", "follows", "for", "former", "formerly", "forth", "four", "from", "further", "furthermore", "g", "get", "gets", "getting", "given", "gives", "go", "goes", "going", "gone", "got", "gotten", "greetings", "h", "had", "hadn't", "happens", "hardly", "has", "hasn't", "have", "haven't", "having", "he", "he's", "hello", "help", "hence", "her", "here", "here's", "hereafter", "hereby", "herein", "hereupon", "hers", "herself", "hi", "him", "himself", "his", "hither", "hopefully", "how", "howbeit", "however", "i", "i'd", "i'll", "i'm", "i've", "ie", "if", "ignored", "immediate", "in", "inasmuch", "inc", "indeed", "indicate", "indicated", "indicates", "inner", "insofar", "instead", "into", "inward", "is", "isn't", "it", "it'd", "it'll", "it's", "its", "itself", "j", "just", "k", "keep", "keeps", "kept", "know", "knows", "known", "l", "last", "lately", "later", "latter", "latterly", "least", "less", "lest", "let", "let's", "like", "liked", "likely", "little", "look", "looking", "looks", "ltd", "m", "mainly", "many", "may", "maybe", "me", "mean", "meanwhile", "merely", "might", "more", "moreover", "most", "mostly", "much", "must", "my", "myself", "n", "name", "namely", "nd", "near", "nearly", "necessary", "need", "needs", "neither", "never", "nevertheless", "new", "next", "nine", "no", "nobody", "non", "none", "noone", "nor", "normally", "not", "nothing", "novel", "now", "nowhere", "o", "obviously", "of", "off", "often", "oh", "ok", "okay", "old", "on", "once", "one", "ones", "only", "onto", "or", "other", "others", "otherwise", "ought", "our", "ours", "ourselves", "out", "outside", "over", "overall", "own", "p", "particular", "particularly", "per", "perhaps", "placed", "please", "plus", "possible", "presumably", "probably", "provides", "q", "que", "quite", "qv", "r", "rather", "rd", "re", "really", "reasonably", "regarding", "regardless", "regards", "relatively", "respectively", "right", "s", "said", "same", "saw", "say", "saying", "says", "second", "secondly", "see", "seeing", "seem", "seemed", "seeming", "seems", "seen", "self", "selves", "sensible", "sent", "serious", "seriously", "seven", "several", "shall", "she",

"should", "shouldn't", "since", "six", "so", "some", "somebody", "somehow", "someone", "something", "sometime", "sometimes", "somewhat", "somewhere", "soon", "sorry", "specified", "specify", "specifying", "still", "sub", "such", "sup", "sure", "t", "t's", "take", "taken", "tell", "tends", "th", "than", "thank", "thanks", "thanx", "that", "that's", "thats", "the", "their", "theirs", "them", "themselves", "then", "thence", "there", "there's", "thereafter", "thereby", "therefore", "therein", "theres", "thereupon", "these", "they", "they'd", "they'll", "they're", "they've", "think", "third", "this", "thorough", "thoroughly", "those", "though", "three", "through", "throughout", "thru", "thus", "to", "together", "too", "took", "toward", "towards", "tried", "tries", "truly", "try", "trying", "twice", "two", "u", "un", "under", "unfortunately", "unless", "unlikely", "until", "unto", "up", "upon", "us", "use", "used", "useful", "uses", "using", "usually", "uucp", "v", "value", "various", "very", "via", "viz", "vs", "w", "want", "wants", "was", "wasn't", "way", "we", "we'd", "we'll", "we're", "we've", "welcome", "well", "went", "were", "weren't", "what", "what's", "whatever", "when", "whence", "whenever", "where", "where's", "whereafter", "whereas", "whereby", "wherein", "whereupon", "wherever", "whether", "which", "while", "whither", "who", "who's", "whoever", "whole", "whom", "whose", "why", "will", "willing", "wish", "with", "within", "without", "won't", "wonder", "would", "wouldn't", "x", "y", "yes", "yet", "you", "you'd", "you'll", "you're", "you've", "your", "yours", "yourself", "yourselves", "z", "zero"

Spanish StopWords (palabras en Español).

"él", "ésta", "éstas", "éste", "éstos", "última", "últimas", "último", "últimos", "a", "añadió", "aún", "actualmente", "adelante", "además", "afirmó", "agregó", "ahí", "ahora", "al", "algún", "algo", "alguna", "algunas", "alguno", "algunos", "alrededor", "ambos", "ante", "anterior", "antes", "apenas", "aproximadamente", "aquí", "así", "aseguró", "aunque", "ayer", "bajo", "bien", "buen", "buena", "buenas", "bueno", "buenos", "cómo", "cada", "casi", "cerca", "cierto", "cinco", "comentó", "como", "con", "conocer", "consideró", "considera", "contra", "cosas", "creo", "cual", "cuales", "cualquier", "cuando", "cuanto", "cuatro", "cuenta", "da", "dado", "dan", "dar", "de", "debe", "deben", "debido", "decir", "dejó", "del", "demás", "dentro", "desde", "después", "dice", "dicen", "dicho", "dieron", "diferente", "diferentes", "dijeron", "dijo", "dio", "donde", "dos", "durante", "e", "ejemplo", "el", "ella", "ellas", "ello", "ellos", "embargo", "en", "encuentra", "entonces", "entre", "era", "eran", "es", "esa", "esas", "ese", "eso", "esos", "está", "están", "esta", "estaba", "estaban", "estamos", "estar", "estará", "estas", "este", "esto", "estos", "estoy", "estuvo", "ex", "existe", "existen", "explicó", "expresó", "fin", "fue", "fuera", "fueron", "gran", "grandes", "ha", "había", "habían", "haber", "habrá", "hace", "hacen", "hacer", "hacerlo", "hacia", "haciendo", "han", "hasta", "hay", "haya", "he", "hecho", "hemos", "hicieron", "hizo", "hoy", "hubo", "igual", "incluso", "indicó", "informó", "junto", "la", "lado", "las", "le", "les", "llegó", "lleva", "llevar", "lo", "los", "luego", "lugar", "más", "manera", "manifestó", "mayor", "me", "mediante", "mejor", "mencionó", "menos", "mi", "mientras", "misma", "mismas", "mismo", "mismos", "momento", "mucha", "muchas", "mucho", "muchos", "muy", "nada", "nadie", "ni", "ningún", "ninguna", "ningunas", "ninguno", "ningunos", "no", "nos", "nosotras", "nosotros", "nuestra", "nuestras", "nuestro", "nuestros", "nueva", "nuevas", "nuevo", "nuevos", "nunca", "o", "ocho", "otra", "otras", "otro", "otros", "para", "parece", "parte", "partir", "pasada", "pasado", "pero", "pesar", "poca", "pocas", "poco", "pocos", "podemos", "podrá", "podrán", "podría", "podrían", "poner", "por", "porque", "posible", "próximo", "próximos", "primer", "primera", "primero", "primeros", "principalmente", "propia", "propias", "propio", "propios", "pudo", "pueda", "puede", "pueden", "pues", "qué", "que", "quedó", "queremos", "quién", "quien", "quienes", "quiere", "realizó", "realizado", "realizar", "respecto", "sí", "sólo", "se", "señaló",

"sea", "sean", "según", "segunda", "segundo", "seis", "ser", "será", "serán", "sería", "si", "sido", "siempre", "siendo", "siete", "sigue", "siguiente", "sin", "sino", "sobre", "sola", "solamente", "solas", "solo", "solos", "son", "su", "sus", "tal", "también", "tampoco", "tan", "tanto", "tenía", "tendrá", "tendrán", "tenemos", "tener", "tenga", "tengo", "tenido", "tercera", "tiene", "tienen", "toda", "todas", "todavía", "todo", "todos", "total", "tras", "trata", "través", "tres", "tuvo", "un", "una", "unas", "uno", "unos", "usted", "va", "vamos", "van", "varias", "varios", "veces", "ver", "vez", "y", "ya", "yo"

French StopWords (palabras en Francés).

"a", "à", "â", "abord", "afin", "ah", "ai", "aie", "ainsi", "allaient", "allo", "allô", "allons", "après", "assez", "attendu", "au", "aucun", "aucune", "aujourd", "aujourd'hui", "auquel", "aura", "auront", "aussi", "autre", "autres", "aux", "auxquelles", "auxquels", "avaient", "avais", "avait", "avant", "avec", "avoir", "ayant", "b", "bah", "beaucoup", "bien", "bigre", "boum", "bravo", "brrr", "c", "ça", "car", "ce", "ceci", "cela", "celle", "celle-ci", "celle-là", "celles", "celles-ci", "celles-là", "celui", "celui-ci", "celui-là", "cent", "cependant", "certain", "certaine", "certaines", "certains", "certes", "ces", "cet", "cette", "ceux", "ceux-ci", "ceux-là", "chacun", "chaque", "cher", "chère", "chères", "chers", "chez", "chiche", "chut", "ci", "cinq", "cinquantaine", "cinquante", "cinquantième", "cinquième", "clac", "clic", "combien", "comme", "comment", "compris", "concernant", "contre", "couic", "crac", "d", "da", "dans", "de", "debout", "dedans", "dehors", "delà", "depuis", "derrière", "des", "dès", "désormais", "desquelles", "desquels", "dessous", "dessus", "deux", "deuxième", "deuxièmement", "devant", "devers", "devra", "différent", "différente", "différentes", "différents", "dire", "divers", "diverse", "diverses", "dix", "dix-huit", "dixième", "dix-neuf", "dix-sept", "doit", "doivent", "donc", "dont", "douze", "douzième", "dring", "du", "duquel", "durant", "e", "effet", "eh", "elle", "elle-même", "elles", "elles-mêmes", "en", "encore", "entre", "envers", "environ", "es", "ès", "est", "et", "etant", "étaient", "étais", "était", "étant", "etc", "été", "etre", "être", "eu", "euh", "eux", "eux-mêmes", "excepté", "f", "façon", "fais", "faisaient", "faisant", "fait", "feront", "fi", "flac", "floc", "font", "g", "gens", "h", "ha", "hé", "hein", "hélas", "hem", "hep", "hi", "ho", "holà", "hop", "hormis", "hors", "hou", "houp", "hue", "hui", "huit", "huitième", "hum", "hurrah", "i", "il", "ils", "importe", "j", "je", "jusqu", "jusque", "k", "l", "la", "là", "laquelle", "las", "le", "lequel", "les", "lès", "lesquelles", "lesquels", "leur", "leurs", "longtemps", "lorsque", "lui", "lui-même", "m", "ma", "maint", "mais", "malgré", "me", "même", "mêmes", "merci", "mes", "mien", "mienne", "miennes", "miens", "mille", "mince", "moi", "moi-même", "moins", "mon", "moyennant", "n", "na", "ne", "néanmoins", "neuf", "neuvième", "ni", "nombreuses", "nombreux", "non", "nos", "notre", "nôtre", "nôtres", "nous", "nous-mêmes", "nul", "o", "o|", "ô", "oh", "ohé", "olé", "ollé", "on", "ont", "onze", "onzième", "ore", "ou", "où", "ouf", "ouias", "oust", "ouste", "outre", "p", "paf", "pan", "par", "parmi", "partant", "particulier", "particulière", "particulièrement", "pas", "passé", "pendant", "personne", "peu", "peut", "peuvent", "peux", "pff", "pfft", "pfut", "pif", "plein", "plouf", "plus", "plusieurs", "plutôt", "pouah", "pour", "pourquoi", "premier", "première", "premièrement", "près", "proche", "psitt", "puisque", "q", "qu", "quand", "quant", "quanta", "quant-à-soi", "quarante", "quatorze", "quatre", "quatre-vingt", "quatrième", "quatrièmement", "que", "quel", "quelconque", "quelle", "quelles", "quelque", "quelques", "quelqu'un", "quels", "qui", "quiconque", "quinze", "quoi", "quoique", "r", "revoici", "revoilà", "rien", "s", "sa", "sacrebleu", "sans", "sapristi", "sauf", "se", "seize", "selon", "sept", "septième", "sera", "seront", "ses", "si", "sien", "sienne", "siennes", "siens", "sinon", "six", "sixième", "soi", "soi-même", "soit", "soixante", "son", "sont", "sous", "stop", "suis", "suivant", "sur", "surtout", "t", "ta", "tac", "tant", "te", "té", "tel", "telle", "tellement", "telles", "tels", "tenant", "tes", "tic",

"tien", "tienne", "tiennes", "tiens", "toc", "toi", "toi-même", "ton", "touchant", "toujours", "tous", "tout", "toute", "toutes", "treize", "trente", "très", "trois", "troisième", "troisièmement", "trop", "tsoin", "tsouin", "tu", "u", "un", "une", "unes", "uns", "v", "va", "vais", "vas", "vé", "vers", "via", "vif", "vifs", "vingt", "vivat", "vive", "vives", "vlan", "voici", "voilà", "vont", "vos", "votre", "vôtre", "vôtres", "vous", "vous-mêmes", "vu", "w", "x", "y", "z", "zut"

German StopWords (palabras en Alemán).

"a", "ab", "aber", "ach", "acht", "achte", "achten", "achter", "achtes", "ag", "alle", "allein", "allem", "allen", "aller", "allerdings", "alles", "allgemeinen", "als", "also", "am", "an", "andere", "anderen", "andern", "anders", "au", "auch", "auf", "aus", "ausser", "außer", "ausserdem", "außerdem", "b", "bald", "bei", "beide", "beiden", "beim", "beispiel", "bekannt", "bereits", "besonders", "besser", "besten", "bin", "bis", "bisher", "bist", "c", "d", "da", "dabei", "dadurch", "dafür", "dagegen", "daher", "dahin", "dahinter", "damals", "damit", "danach", "daneben", "dank", "dann", "daran", "darauf", "daraus", "darf", "darfst", "darin", "darüber", "darum", "darunter", "das", "dasein", "daselbst", "dass", "daß", "dasselbe", "davon", "davor", "dazu", "dazwischen", "dein", "deine", "deinem", "deiner", "dem", "dementsprechend", "demgegenüber", "demgemäss", "demgemäß", "demselben", "demzufolge", "den", "denen", "denn", "denselben", "der", "deren", "derjenige", "derjenigen", "dermassen", "dermaßen", "derselbe", "derselben", "des", "deshalb", "desselben", "dessen", "deswegen", "d.h", "dich", "die", "diejenige", "diejenigen", "dies", "diese", "dieselbe", "dieselben", "diesem", "diesen", "dieser", "dieses", "dir", "doch", "dort", "drei", "drin", "dritte", "dritten", "dritter", "drittes", "du", "durch", "durchaus", "dürfen", "dürft", "durfte", "durften", "e", "eben", "ebenso", "ehrlich", "ei", "eigen", "eigene", "eigenen", "eigener", "eigenes", "ein", "einander", "eine", "einem", "einen", "einer", "eines", "einige", "einigen", "einiger", "einiges", "einmal", "eins", "elf", "en", "ende", "endlich", "entweder", "er", "ernst", "erst", "erste", "ersten", "erster", "erstes", "es", "etwa", "etwas", "euch", "f", "früher", "fünf", "fünfte", "fünften", "fünfter", "fünftes", "für", "g", "gab", "ganz", "ganze", "ganzen", "ganzer", "ganzes", "gar", "gedurft", "gegen", "gegenüber", "gehabt", "gehen", "geht", "gekannt", "gekonnt", "gemacht", "gemocht", "gemusst", "genug", "gerade", "gern", "gesagt", "geschweige", "gewesen", "gewollt", "geworden", "gibt", "ging", "gleich", "gott", "gross", "groß", "grosse", "große", "grossen", "großen", "grosser", "großer", "grosses", "großes", "gut", "gute", "guter", "gutes", "h", "habe", "haben", "habt", "hast", "hat", "hatte", "hätte", "hatten", "hätten", "heisst", "her", "heute", "hier", "hin", "hinter", "hoch", "i", "ich", "ihm", "ihn", "ihnen", "ihr", "ihre", "ihrem", "ihren", "ihrer", "ihres", "im", "immer", "in", "indem", "infolgedessen", "ins", "irgend", "ist", "j", "ja", "jahr", "jahre", "jahren", "je", "jede", "jedem", "jeden", "jeder", "jedermann", "jedermanns", "jedoch", "jemand", "jemandem", "jemanden", "jene", "jenem", "jenen", "jener", "jenes", "jetzt", "k", "kam", "kann", "kannst", "kaum", "kein", "keine", "keinem", "keinen", "keiner", "kleine", "kleinen", "kleiner", "kleines", "kommen", "kommt", "können", "könnt", "konnte", "könnte", "konnten", "kurz", "l", "lang", "lange", "leicht", "leide", "lieber", "los", "m", "machen", "macht", "machte", "mag", "magst", "mahn", "man", "manche", "manchem", "manchen", "mancher", "manches", "mann", "mehr", "mein", "meine", "meinem", "meinen", "meiner", "meines", "mensch", "menschen", "mich", "mir", "mit", "mittel", "mochte", "möchte", "mochten", "mögen", "möglich", "mögt", "morgen", "muss", "muß", "müssen", "musst", "müsst", "musste", "mussten", "n", "na", "nach", "nachdem", "nahm", "natürlich", "neben", "nein", "neue", "neuen", "neun", "neunte", "neunten", "neunter", "neuntes", "nicht", "nichts", "nie", "niemand", "niemandem", "niemanden", "noch", "nun", "nur", "o", "ob", "oben", "oder", "offen", "oft", "ohne", "ordnung", "p", "q", "r", "recht", "rechte", "rechten", "rechter", "rechtes", "richtig", "rund", "s",

"sa", "sache", "sagt", "sagte", "sah", "satt", "schlecht", "schluss", "schon", "sechs", "sechste", "sechsten", "sechster", "sechstes", "sehr", "sei", "seid", "seien", "sein", "seine", "seinem", "seinen", "seiner", "seines", "seit", "seitdem", "selbst", "sich", "sie", "sieben", "siebente", "siebenten", "siebenter", "siebentes", "sind", "so", "solang", "solche", "solchem", "solchen", "solcher", "solches", "soll", "sollen", "sollte", "sollten", "sondern", "sonst", "sowie", "später", "statt", "t", "tag", "tage", "tagen", "tat", "teil", "tel", "tritt", "trotzdem", "tun", "u", "über", "überhaupt", "übrigens", "uhr", "um", "und", "uns", "unser", "unsere", "unserer", "unter", "v", "vergangenen", "viel", "viele", "vielem", "vielen", "vielleicht", "vier", "vierte", "vierten", "vierter", "viertes", "vom", "von", "vor", "w", "wahr?", "während", "währenddem", "währenddessen", "wann", "war", "wäre", "waren", "wart", "warum", "was", "wegen", "weil", "weit", "weiter", "weitere", "weiteren", "weiteres", "welche", "welchem", "welchen", "welcher", "welches", "wem", "wen", "wenig", "wenige", "weniger", "weniges", "wenigstens", "wenn", "wer", "werde", "werden", "werdet", "wessen", "wie", "wieder", "will", "willst", "wir", "wird", "wirklich", "wirst", "wo", "wohl", "wollen", "wollt", "wollte", "wollten", "worden", "wurde", "würde", "wurden", "würden", "x", "y", "z", "z.b", "zehn", "zehnte", "zehnten", "zehnter", "zehntes", "zeit", "zu", "zuerst", "zugleich", "zum", "zunächst", "zur", "zurück", "zusammen", "zwanzig", "zwar", "zwei", "zweite", "zweiten", "zweiter", "zweites", "zwischen", "zwölf"

Italian StopWords (palabras en Italiano).

"a", "abbastanza", "abbiomo", "accidenti", "ad", "adesso", "affinche", "affinché", "agli", "ahime", "ahimè", "ai", "al", "alcuna", "alcuni", "alcuno", "all", "alla", "alle", "allo", "altri", "altrimenti", "altro", "altrui", "anche", "ancora", "anni", "anno", "ansa", "assai", "attesa", "avanti", "avendo", "avente", "aver", "avere", "avete", "aveva", "avuta", "avute", "avuti", "avuto", "basta", "bene", "benissimo", "berlusconi", "brava", "bravo", "c", "casa", "caso", "cento", "certa", "certe", "certi", "certo", "che", "chi", "chicchessia", "chinque", "chiunque", "ci", "ciascuna", "ciascuno", "cima", "cio", "ciò", "cioe", "cioè", "circa", "citta", "città", "codesta", "codeste", "codesti", "codesto", "cogli", "coi", "col", "colei", "coll", "coloro", "colui", "come", "con", "concernente", "consiglio", "contro", "cortesia", "cos", "cosa", "cosi", "cosí", "così", "cui", "d", "da", "dagli", "dai", "dal", "dall", "dalla", "dalle", "dallo", "davanti", "degli", "dei", "del", "dell", "della", "delle", "dello", "dentro", "detto", "deve", "di", "dice", "dietro", "dila", "dire", "dirimpetto", "dopo", "dove", "dovra", "dovrà", "due", "dunque", "durante", "e", "è", "ecco", "ed", "egli", "ella", "eppure", "era", "erano", "esse", "essendo", "esser", "essere", "essi", "ex", "fa", "fare", "fatto", "favore", "fin", "finalmente", "finche", "finché", "fine", "fino", "forse", "fra", "frattanto", "fuori", "gia", "già", "giacche", "giacché", "giorni", "giorno", "gli", "gliela", "gliele", "glieli", "glielo", "gliene", "governo", "grande", "grazie", "gruppo", "ha", "hai", "hanno", "ho", "i", "ieri", "il", "improvviso", "in", "infatti", "insieme", "intanto", "intorno", "invece", "invere", "io", "l", "la", "là", "lavoro", "le", "lei", "li", "lo", "lontano", "loro", "lui", "lungo", "ma", "macche", "macché", "magari", "mai", "male", "malgrado", "malissimo", "me", "medesimo", "mediante", "meglio", "meno", "mentre", "mesi", "mezzo", "mi", "mia", "mie", "miei", "mieri", "mila", "miliardi", "milioni", "ministro", "mio", "moltissimo", "molto", "mondo", "nazionale", "ne", "né", "negli", "nei", "nel", "nell", "nella", "nelle", "nello", "nemmeno", "neppure", "nessuna", "nessuno", "niente", "no", "noi", "non", "nondimeno", "nondimento", "nostra", "nostre", "nostri", "nostro", "nulla", "nuovo", "o", "od", "oggi", "ogni", "ognuna", "ognuno", "oltre", "oppure", "ora", "ore", "osi", "osì", "ossia", "paese", "parecchi", "parecchie", "parecchio", "parte", "partendo", "peccato", "peggio", "per", "perche", "perché", "perchè", "percio", "perciò", "perfino", "pero", "però", "perque", "perqué", "persone", "piedi", "pieno", "piglia", "piu", "piú", "più", "po",

"pochissimo", "poco", "poi", "poiche", "poiché", "press", "prima", "primo", "proprio", "puo", "può", "pure", "purtroppo", "qualche", "qualcuna", "qualcuno", "quale", "quali", "qualsiani", "qualunque", "quando", "quanta", "quante", "quanti", "quanto", "quantunque", "quasi", "quattro", "quel", "quella", "quelli", "quello", "quest", "questa", "queste", "questi", "questo", "qui", "quindi", "riecco", "rieccò", "saltro", "salvo", "sara", "sarà", "sarebbe", "scopo", "scorso", "se", "sé", "secondo", "seguente", "sei", "sempre", "senza", "si", "sí", "sia", "siamo", "siete", "solito", "solo", "sono", "soppra", "sopra", "sotto", "sta", "staranno", "stata", "state", "stati", "stato", "stesso", "stresso", "su", "sua", "successivo", "sue", "sugli", "sui", "sul", "sull", "sulla", "sulle", "sullo", "suo", "suoi", "tale", "talvolta", "tanto", "te", "tempo", "ti", "torino", "tra", "tranne", "trannefino", "tre", "troppo", "tu", "tua", "tue", "tuo", "tuoi", "tutta", "tuttavia", "tutte", "tutti", "tutto", "uguali", "un", "una", "uno", "uomo", "uori", "va", "vale", "varia", "varie", "vario", "verso", "vi", "via", "vicino", "vise", "visé", "visto", "vita", "voi", "volta", "vostra", "vostre", "vostri", "vostro"

Portuguese StopWords (palabras en Portugués).

"a", "à", "adeus", "agora", "aí", "ainda", "além", "algo", "algumas", "alguns", "ali", "ano", "anos", "antes", "ao", "aos", "apenas", "apoio", "após", "aquela", "aquelas", "aquele", "aqueles", "aqui", "aquilo", "área", "as", "às", "assim", "até", "atrás", "através", "baixo", "bastante", "bem", "bom", "breve", "cá", "cada", "catorze", "cedo", "cento", "certamente", "certeza", "cima", "cinco", "coisa", "com", "como", "conselho", "contra", "custa", "da", "dá", "dão", "daquela", "daquele", "dar", "das", "de", "debaixo", "demais", "dentro", "depois", "desde", "dessa", "desse", "desta", "deste", "deve", "deverá", "dez", "dezanove", "dezasseis", "dezassete", "dezoito", "dia", "diante", "diz", "dizem", "dizer", "do", "dois", "dos", "doze", "duas", "dúvida", "e", "é", "ela", "elas", "ele", "eles", "em", "embora", "entre", "era", "és", "essa", "essas", "esse", "esses", "esta", "está", "estar", "estas", "estás", "estava", "este", "estes", "esteve", "estive", "estivemos", "estiveram", "estiveste", "estivestes", "estou", "eu", "exemplo", "faço", "falta", "favor", "faz", "fazeis", "fazem", "fazemos", "fazer", "fazes", "fez", "fim", "final", "foi", "fomos", "for", "foram", "forma", "foste", "fostes", "fui", "geral", "grande", "grandes", "grupo", "há", "hoje", "horas", "isso", "isto", "já", "lá", "lado", "local", "logo", "longe", "lugar", "maior", "maioria", "mais", "mal", "mas", "máximo", "me", "meio", "menor", "menos", "mês", "meses", "meu", "meus", "mil", "minha", "minhas", "momento", "muito", "muitos", "na", "nada", "não", "naquela", "naquele", "nas", "nem", "nenhuma", "nessa", "nesse", "nesta", "neste", "nível", "no", "noite", "nome", "nos", "nós", "nossa", "nossas", "nosso", "nossos", "nova", "nove", "novo", "novos", "num", "numa", "número", "nunca", "o", "obra", "obrigada", "obrigado", "oitava", "oitavo", "oito", "onde", "ontem", "onze", "os", "ou", "outra", "outras", "outro", "outros", "para", "parece", "parte", "partir", "pela", "pelas", "pelo", "pelos", "perto", "pode", "pôde", "podem", "poder", "põe", "põem", "ponto", "pontos", "por", "porque", "porquê", "posição", "possível", "possivelmente", "posso", "pouca", "pouco", "primeira", "primeiro", "próprio", "próximo", "puderam", "qual", "quando", "quanto", "quarta", "quarto", "quatro", "que", "quê", "quem", "quer", "quero", "questão", "quinta", "quinto", "quinze", "relação", "sabe", "são", "se", "segunda", "segundo", "sei", "seis", "sem", "sempre", "ser", "seria", "sete", "sétima", "sétimo", "seu", "seus", "sexta", "sexto", "sim", "sistema", "sob", "sobre", "sois", "somos", "sou", "sua", "suas", "tal", "talvez", "também", "tanto", "tão", "tarde", "te", "tem", "têm", "temos", "tendes", "tenho", "tens", "ter", "terceira", "terceiro", "teu", "teus", "teve", "tive", "tivemos", "tiveram", "tiveste", "tivestes", "toda", "todas", "todo", "todos", "trabalho", "três", "treze", "tu", "tua", "tuas", "tudo", "um", "uma", "umas", "uns", "vai", "vais", "vão", "vários",

"vem", "vêm", "vens", "ver", "vez", "vezes", "viagem", "vindo", "vinte", "você", "vocês", "vos", "vós", "vossa", "vossas", "vosso", "vossos", "zero"

2. LISTA DE VERBOS

A continuación se presenta la lista de verbos en inglés que se utilizó para complementar la fase de etiquetado del algoritmo IGBHSK, la cual se tomo de los sitios "Vocabulix", específicamente de la página web "Verbos en Inglés" (http://www.vocabulix.com/conjugacion/Verbos-Ingles.html), y también se tuvo en cuenta el sitio "Speakspeak Better English", específicamente de la página web "Lista de verbos irregulares en Inglés / Los participios pasados" (http://www.speakspeak.com/html/d2f_resources_irregulares_verbos_ingles_es.htm).

English Verbs (Verbos en Inglés)

"accompany", "accustom", "act", "add", "address", "advertise", "agree", "aid", "amuse", "annoy", "answer", "appeal", "appear", "approach", "arise", "arrange", "arrest", "arrive", "ask", "assist", "attend", "awake", "balance", "banish", "bark", "bear", "beat", "become", "beg", "begin", "behave", "believe", "belong", "bend", "bet", "bind", "bite", "bless", "blow", "board", "boil", "break", "breathe", "bring", "brush", "build", "burn", "burst", "buy", "call", "care", "carry", "catch", "change", "charge", "check", "choose", "clean", "climb", "cling", "close", "comb", "come", "complete", "consist", "cook", "cost", "count", "cover", "crash", "crawl", "creep", "cross ", "cry", "cut", "dance", "deal", "declare", "delay", "deliver", "deny", "dial", "die", "dig", "dine", "do", "draw", "dress", "drink", "drive", "drop", "dry", "enclose", "engage", "enjoy", "envy", "exclaim", "explain", "express", "fail", "fall", "fasten", "feed", "feel", "fight", "file", "fill", "find", "find out", "finish", "fire", "fish", "fix", "flee", "fly", "follow", "forbid", "foresee", "forget", "forgive", "freeze", "frighten", "fry", "gain", "get", "give", "go", "grind", "grow", "guess", "hang", "happen", "have", "hear", "help", "hide", "hit", "hold", "hope", "hurry", "hurt", "imagine", "iron", "judge", "keep", "kill", "kiss", "know", "laugh", "lay", "lead", "leak", "lean", "learn", "leave", "lend", "let", "lie", "light", "like", "lock", "look", "lose", "make", "manage", "mark", "marry", "massage", "mean", "measure", "meet", "melt", "milk", "miss", "mistake", "misunderstand", "move", "observe", "offer", "open", "order", "observe", "overcome", "park", "pass", "pay", "perform", "phone", "pick", "plan", "play", "please", "plough", "polish", "pour", "practice", "pray", "prefer", "prepare", "promise", "pronounce", "pull", "punish", "push", "put", "rain", "raise", "reach", "read", "realize", "rebuild", "receive", "refuse", "register", "Remain", "Remember", "Repair", "repeat", "report", "request", "Require", "Reserve", "Resolve", "rest", "Return", "rid", "ride", "ring", "rise", "Row", "run", "Save", "saw", "say", "Search", "see", "seek", "sell", "send", "set", "settle", "shake", "shed", "shine", "shoot", "show", "shrink", "shut", "sign", "sing", "sink", "sit", "sleep", "slide", "slip", "smell", "smile", "smoke", "snow", "speak", "speed", "spend", "spill", "spin", "split", "spoil", "spread", "spring", "stand", "stay", "steal", "stick", "stink", "stop", "stretch", "strike", "study", "suffer", "swallow", "swell", "swim", "swing", "switch", "take", "talk", "teach", "tear", "tell", "thank", "think", "throw", "thrust", "tire", "touch", "train", "trap", "travel", "trouble", "try", "turn", "undergo", "understand", "undertake", "undo", "unpack", "use", "visit", "wait", "wake", "walk", "want", "warm", "warn", "wash", "watch", "water", "wear", "weigh", "whistle", "win", "wind", "wish", "withdraw", "withstand", "work", "wrap up", "wreck", "write".

ANEXO B – ALGORITMO DE PORTER

3. ALGORITMO DE PORTER

El algoritmo de Porter[1] [1] es un proceso para eliminar los sufijos comunes morfológicos e inflexiones de las palabras en inglés. Se utiliza como un proceso de normalización que usualmente se realiza para establecer un Sistema de Recuperación de Información. El artículo original del algoritmo (*An algorithm for suffix stripping*) lo escribió Martin F. Porter en 1979 en el Laboratorio de Computación de Cambridge (Inglaterra), como parte de un proyecto de Recuperación de Información. Actualmente existen algoritmos de Porter especializados a otros lenguajes, entre ellos el español.

A continuación se presenta el algoritmo de Porter [1], el cual fue implementado en Lucene.net, cuya implementación, a su vez fue utilizada en la fase de pre-procesamiento del Algoritmo Iterativo de la Mejor Búsqueda Armónica Global y K-means (IGBHSK) propuesto en la presente tesis de grado.

3.1. INTRODUCCIÓN

La eliminación de sufijos por medios automáticos es una operación que es especialmente útil en el campo de la Recuperación de la Información (IR). En un ambiente típico de Recuperación de Información un documento está representado por un vector de palabras, o términos. Los términos con una raíz común normalmente tienen significado similar, por ejemplo:

CONNECT
CONNECTED
CONNECTING
CONNECTION
CONNECTION

Frecuentemente, el desempeño de un sistema de IR se mejorará si los grupos de términos como éstos son combinados dentro de un solo término. Esto se puede hacer por la eliminación de varios sufijos –ED, -ING, -ION, -IONS para dejar solo el término CONNECT. Además, el proceso de eliminación reducirá el número total de términos en el sistema de IR, y por lo tanto reduce el tiempo y la complejidad de los datos en el sistema, lo cual es siempre conveniente.

Muchas estrategias para la eliminación de sufijos se han reportado en la literatura. La naturaleza de la tarea variará considerablemente dependiendo de si se utiliza un diccionario de raíces, o una lista de sufijos, y por supuesto del propósito para el cual el eliminador de sufijos se ha hecho. Asumiendo que no se hace uso de un diccionario de raíces, y que el propósito de la tarea es mejorar el desempeño del sistema de IR, el programa de eliminación de sufijos normalmente dará una lista explícita de sufijos, y, con cada sufijo, el criterio bajo el cual este puede ser eliminado de una palabra para dejar una raíz válida.

Los principales aspectos positivos del algoritmo de Porter son: es pequeño, rápido y sencillo.

[1] Traducción libre del artículo escrito por Martin Porter, titulado "*An algorithm for suffix stripping".*

3.2. PASOS DEL ALGORITMO

El algoritmo de Porter trabaja efectivamente tratando sufijos complejos así como sufijos compuestos formados por sufijos simples, y eliminando los sufijos simples en una serie de pasos. En cada paso la eliminación que se hace del sufijo depende de la raíz restante, que normalmente involucra una medida de su longitud se sílabas.

Para presentar complemente el algoritmo de eliminación de sufijos son necesarias algunas definiciones que se presentan a continuación:

Una *consonante* en una palabra es una letra diferente a A, E, I, O y U, y distinta a Y precedida por una consonante. De esta manera, en TOY las consonantes son T y Y, y en SYZYGY las consonantes son S, Z y G. Si una letra no es una consonante es una vocal.

Una consonante será denotada por *c*, una vocal por *v*. Una lista ccc… de longitud más grande que 0 será denotada por C, y una lista vvv… de longitud más grande que 0 será denotada por V. Cualquier palabra, o parte de una palabra, por lo tanto tiene una de las cuatro formas:

CVCV…C
CVCV…V
VCVC…C
VCVC…V

Las cuales pueden ser representadas, a su vez, por una sola forma:

[C] VCVC …[V]

Donde los paréntesis cuadrados denotan presencia arbitraria de sus contenidos. Usando $(VC)^m$ para denotar VC repetido m veces, esto puede de nuevo ser escrito como:

$[C](VC)^m[V]$.

m será llamado la medida de cualquier palabra o parte de una palabra cuando se represente de esta forma. El caso *m* = 0 cubre la palabra nula. Aquí se presentan algunos ejemplos:

m = 0 TR, EE, TREE, Y, BY.
m = 1 TROUBLE, OATS, TREES, IVY.
m = 2 TROUBLES, PRIVATE, OATEN, ORRERY.

Las reglas para la eliminación de sufijos serán dadas en la forma:

(Condición) S1 -> S2

Esto significa que si una palabra finaliza con el sufijo S1, y la raíz antes de S1 satisface la condición dada, S1 es reemplazada por S2. La condición está normalmente dada en términos de m, por ejemplo:

(*m* > 1) EMENT ->

Aquí S1 es “EMENT” y S2 es nulo. Esto convertiría REPLACEMENT a REPLAC, puesto que REPLAC es un parte de la palabra para la que *m*=2.
La parte de la condición puede también contener lo siguiente:

*S – la raíz finaliza con S (y similarmente para las otras letras)
v- la raíz contiene una vocal.
*d – la raíz finaliza con una consonante doble (por ejemplo, -TT, -SS).
*o – la raíz finaliza con cvc, donde la segunda c no es W, X, o Y (por ejemplo – WIL, -HOP).

Y la parte de la condición también puede contener expresiones con and, or y not, de modo que:

(*m* > 1 and (*S or *T))

Prueba una raíz con m > 1 que finalice en S o T, mientras,

(*d and not (*L or *S or *Z))

Prueba una raíz que finalice con una consonante doble distinta de L, S o Z. Las condiciones elaboradas como esta son requeridas muy raramente.

En un conjunto de reglas escritas unas tras otras, solo una se cumple, y esta será una con la correspondencia más larga de S1 para una palabra dada. Por ejemplo, con:

SSES -> SS
IES -> I
SS -> SS
S ->

(Aquí las condiciones son todas nulas) CARESSES corresponde a CARESS puesto que SSES es la coincidencia más larga para S1. Igualmente CARESS corresponde a CARESS (S1 = “SS”) y CARES a CARE (S1 = “S”).

El algoritmo presenta cinco pasos, los cuales están compuestos por reglas que se aplican a las palabras de las que se obtendrá la raíz. En estas reglas, los ejemplos de su aplicación (con éxito o no) se presentan a la derecha en minúsculas.

Paso 1a

SSES -> SS	caresses -> caress
IES -> I	ponies ->poni
	Ties -> ti
SS -> SS	caress -> caress
S ->	cats -> cat

Paso 1b

(m > 0) EED -> EE	feed -> feed
	agreed -> agree
(*v*) ED ->	plastered -> plaster
	Bled -> bled
(*v*) ING ->	motoring -> motor
	Sing -> sing

Si la segunda o tercera de las reglas en el Paso 1b es exitoso, se hace lo siguiente:

AT -> ATE	conflat(ed) -> conflate
BL -> BLE	troubl(ing) -> trouble
IZ -> IZE	siz(ed) -> size

(*d and not (*L or *S or *Z)) ->sola letra

hopping (ing) -> hop
Tann(ed) -> tan
Fall(ing) -> fall
Hiss(ing) -> hiss
Fizz(ed) -> fizz

(m=1 and *o) -> E

fail(ing) -> fail
Fil(ing) -> file

La regla para mapear a una sola letra provoca la eliminación de una letra doble. El –E es puesto en –AT, -BL y –IZ, de modo que los sufijos –ATE, -BLE y –IZE pueden ser reconocidos después. Esta E puede ser eliminada en el paso 4.

Paso 1c

(*v*) Y -> I	happy -> happi
	Sky -> sky

El paso 1 trata con plurales y pasados participios. Los pasos siguientes son mucho más sencillos.

Paso 2

(m>0) ATIONAL -> ATE	relational -> relate
(m>0) TIONAL -> TION	conditional ->condition
	Rational -> rational
(m>0)ENCI ->ENCE	valenci -> valence
(m>0) ANCI -> ANCE	hesitanci ->hesitance
(m>0) IZER -> IZE	digitizer -> digitize
(m>0) ABLI -> ABLE	conformabli -> conformable
(m>0) ALLI -> AL	radicalli -> radical
(m>0) ENTLI -> ENT	differentli -> different

(m>0) ELI -> E	vileli -> vile
(m>0) OUSLI ->OUS	analogousli -> analogous
(m>0) IZATION -> IZE	vietnamization ->vietnamize
(m>0) ATION ->ATE	predication -> predicate
(m>0) ATOR -> ATE	operator -> operate
(m>0) ALISM ->AL	feudalism -> feudal
(m>0) IVENESS -> IVE	dicisiveness -> decisive
(m>0) FULNESS -> FUL	hopefulness - > hopeful
(m>0) OUSNESS -> OUS	callousness -> callous
(m>0) ALITI -> AL	formaliti -> formal
(m>0) IVITI -> IVE	sensitivity -> sensitive
(m>0)BILITI -> BLE	sensibility -> sensible

La prueba para la cadena S1 se puede hacer rápido haciendo un programa que cambie la penúltima letra de la palabra que está siendo probada. Esto da una descomposición bastante justa de los posibles valores de la cadena S1. Esto se verá en el hecho de que las cadenas S1 en el paso 2 son presentadas en el orden alfabético de la penúltima letra. Técnicas similares pueden ser aplicadas en los otros pasos.

Paso 3

(m>0) ICATE -> IC	triplícate -> triplic
(m>0) ATIVE ->	formative -> form
(m>0) ALIZE ->AL	formalize -> formal
(m>0) ICITI -> IC	electriciti -> electric
(m>0) ICAL -> IC	electrical -> electric
(m>0) FUL ->	hopeful -> hope
(m>0) NESS ->	goodness -> good

Paso 4

(m > 1) AL ->	revival -> reviv
(m > 1) ANCE ->	allowance -> allow
(m > 1) ENCE ->	inference -> infer
(m > 1) ER ->	airliner - > airlin
(m > 1) IC ->	gyroscopic -> gyroscop
(m > 1) ABLE ->	adjustable -> adjust
(m > 1) IBLE ->	defensible -> defens
(m > 1) ANT ->	irritant -> irrit
(m > 1) EMENT ->	replacement -> replac
(m > 1) ENT ->	dependent -> depend
(m > 1) and (*S or *T)ION ->	adoption -> adop
(m > 1) OU ->	homologou -> homolog
(m > 1)ISM ->	communism -> commun
(m > 1) ATE ->	activate -> activ
(m > 1) ITI ->	angulariti -> angular
(m > 1)OUS ->	homologous -> homolog
(m > 1) IVE ->	effective -> effect
(m > 1)IZE ->	bowdlerize -> bowdler

Los sufijos son ahora eliminados. El resto es un poco arreglado.

Paso 5a

(m > 1) E ->	probate -> probat
	Rate -> rate
(m = 1 and not *o)E ->	cease -> ceas

Paso 5b

(m > 1 and *d and *L) ->	una sola letra
	Controll -> control
	Roll -> roll

El algoritmo es cuidadoso de no eliminar un sufijo cuando la raíz es demasiado corta, la longitud de la raíz está dada por su medida m. No hay una base lingüística para este enfoque. Solo fue observado que m podría ser bastante efectivo para ayudar a decidir si es o no prudente quitar un sufijo.

Lista A	Lista B
RELATE	DERIVATE
PROBATE	ACTIVATE
CONFLATE	DEMONSTRATE
PIRATE	NECESSITATE
PRELATE	RENOVATE

-ATE es eliminado de las palabras de la lista B, pero no de las palabras de lista A. Esto significa que los pares DERIVATE/DERIVE, ACTIVATE/ACTIVE, DEMOSTRATE/DEMOSTRABLE, NECESSITATE/NECESSITOUS, se combinarán a la vez. El hecho de que no se hace ningún esfuerzo para identificar los prefijos puede hacer que los resultados parezcan más bien incoherentes. Por lo tanto PRELATE no pierde el –ATE, pero ARCHPRELATE se vuelve ARCHPREL. En la práctica esto no es demasiado importante, porque la presencia del prefijo disminuye la probabilidad de una combinación errónea.

Los sufijos complejos son eliminados parte por parte en los diferentes pasos. De esta manera, GENERALIZATIONS es reducido a GENERALIZATION (paso 1), luego a GENERALIZE (paso 2), luego a GENERAL (paso 3), y luego a GENER (paso 4). OSCILLATORS es reducido a OSCILLATOR (paso 1), luego a OSCILLATE (paso 2), luego a OSCILL (paso 4), y luego a OSCIL (paso 5). En un vocabulario de 10.000 palabras, la reducción en el tamaño de la raíz fue distribuida entre los pasos como sigue:

La eliminación de sufijos de un vocabulario de 10.000 palabras

Número de palabras reducidas en el paso 1: 3597
Número de palabras reducidas en el paso 2: 766
Número de palabras reducidas en el paso 3: 327
Número de palabras reducidas en el paso 4: 2424

Número de palabras reducidas en el paso 5: 1373
Número de palabras no reducidas: 3650

El vocabulario resultante de las raíces contenía 6370 entradas distintas. De esta manera el proceso de eliminación redujo el tamaño del vocabulario aproximadamente en un tercio.

ANEXO C – ALGORITMO FPGROWTH

4. ALGORITMO FPGROWTH

4.1. INTRODUCCIÓN

Este algoritmo está basado en una representación de árbol de prefijos de una base de datos de transacciones llamada Frequent Pattern Tree (Árbol de Patrones Frecuentes) [2]. "La idea básica del algoritmo FP-Growth puede ser descrita como un esquema de eliminación recursiva: en un primer paso de pre-procesamiento se borran todos los ítems de las transacciones que no son frecuentes individualmente o no aparecen en el mínimo soporte de transacciones, luego se seleccionan todas las transacciones que contienen al menos un ítem frecuente, se realiza esto de manera recursiva hasta obtener una base de datos reducida. Al retorno, se remueven los ítems procesados de la base datos de transacciones en la memoria y se empieza otra vez, y así con el siguiente ítem frecuente" [3].

"Los ítems en cada transacción son almacenados y luego se ordena descendentemente su frecuencia en la base de datos. Después de que se han borrado todos los ítems infrecuentes de la base de datos de transacciones, se pasa al árbol FP. Un árbol FP es básicamente de prefijos para las transacciones, esto es: cada camino representa el grupo de transacciones que comparten el mismo prefijo, cada nodo corresponde a un ítem. Todos los nodos que referencian al mismo ítem son referenciados juntos en una lista, de modo que todas las transacciones que contienen un ítem específico pueden encontrarse fácilmente y contarse al recorrer la lista. Esta lista puede ser accedida a través de la cabeza, lo cual también expone el número total de ocurrencias del ítem en la base de datos" [3].

Inserciones en la base de datos: este algoritmo no requiere de la generación de candidatos, por lo tanto, precisa de pocos accesos a la base de datos [4].

Costo computacional: el algoritmo está basado en una representación de árbol de prefijos de una base de datos de transacciones, por lo tanto no necesita de la creación de un árbol de prefijos; sin embargo, la creación de dicho árbol no requiere de un costo computacional elevado [5].

Tiempo de ejecución: este algoritmo busca patrones frecuentes con una corta búsqueda recursiva de prefijos, lo que en tiempo de ejecución es muy superior al del A priori, ya que no requiere de constantes accesos a la base de datos [4].

Rendimiento: puede generar un árbol FP-Tree de una base de datos proyectada si el árbol inicial no se puede alojar completamente en la memoria principal, lo que le permite adecuarse a los recursos disponibles [5].

De acuerdo a lo anterior, se decide implementar el algoritmo FPGrowth ya que tiene ventajas operacionales sobre los otros al no necesitar de la generación de ítems candidatos y ser computacionalmente más rápido. Entre las razones por la que se seleccionó este algoritmo tenemos que requiere de pocos accesos a la base. Este algoritmo está basado en una representación de árbol de prefijos de una base de datos de transacciones; sin embargo, la creación de dicho árbol no requiere de un costo

computacional elevado. El algoritmo busca patrones frecuentes con una corta búsqueda recursiva de prefijos, lo que en tiempo de ejecución es muy superior al A priori, ya que no requiere constantes accesos a la base de datos [5].

4.2. PASOS DEL ALGORITMO

El algoritmo FP-growth está compuesto de dos pasos importantes, que a su vez se dividen de otros pasos más específicos. En el primero gran paso se debe construir un Árbol de Patrones Frecuente o FP-tree y en el segundo paso se desarrolla un método basado en el FP-tree creado en el paso anterior, el método es llamado Crecimiento de Patrones Frecuentes (Frequent Patter-growth) ó FP-growth, el cual descubre los conjuntos completos de patrones frecuentes [2]. A continuación se describen estos dos pasos importantes en el algoritmo FP-growth[2].

1. Diseño y construcción de un árbol de patrones frecuentes ó FP-tree

Definición: Un FP-tree es un árbol de patrones frecuentes, cuya estructura de árbol se define como sigue:

1. Consiste de una raíz etiquetada como “null”, un conjunto de sub-árboles de prefijos de ítems como el hijo de la raíz, y una tabla de cabecera de ítems frecuentes.

2. Cada nodo en el sub-árbol de prefijos de ítem consiste de tres campos: nombre de ítem, cuenta, y un enlace de nodo, donde el nombre del ítem registra el ítem que el nodo representa, la cuenta registra el número de transacciones representadas por la porción del camino que alcanza este nodo, y el enlace de nodo enlaza al siguiente nodo en el FP-tree que lleva el mismo nombre de ítem, o null si no hay uno.

3. Cada entrada en la tabla de cabecera de ítems frecuentes consiste de dos campos, (1) el nombre del ítem y (2) la cabeza del enlace de nodo, la cual apunta al primer nodo en el FP-tree que lleva el nombre del ítem.

Con base en esta definición, se tiene el siguiente algoritmo de construcción de FP-tree.

Algoritmo 1: Construcción del Árbol FP-tree

Entrada: Una base de datos de transacciones *DB* y un umbral de soporte mínimo ξ.
Salida: Su árbol de patrones frecuentes, FP-tree.

Método: El FP-tree es construido en los siguientes pasos.

1. Escanear la base de datos *DB* una vez. Reunir el conjunto de ítems frecuentes F y sus soportes. Ordenar F en orden descendente de soporte como L, la lista de ítems frecuentes.

[2] Traducción libre del artículo escrito por Han J. titulado “Mining Frequent Patterns without Candidate Generation” [2].

2. Crear la raíz de un FP-tree, *T*, y etiquetarlo como "*null*". Para cada transacción *Trans* en *DB* hacer lo siguiente.

Seleccionar y ordenar los ítems frecuentes en *Trans* de acuerdo al orden de *L*. Permitir que la lista ordenada de ítems frecuentes en *Trans* sea *[p | P]*, donde *p* es el primer elemento y *P* es la lista restante. Llamar a la función *insert_tree([p | P],T)*.

La función *insert_tree([p | P],T)* se ejecuta como sigue. Si *T* tiene un hijo *N* tal que *N.item-name = p.item-name*, entonces incrementar la cuenta de *N* en 1; sino crear un nuevo nodo N, y permitir que su cuenta sea 1, su enlace de padre esté enlazado a *T*, y su enlace de nodo esté enlazado a los nodos con el mismo *ítem-name* por medio de la estructura de enlace de nodo. Si *P* no está vacío, llamar a *insert-tree(P,N)* recursivamente.

Desde el proceso de construcción del FP-tree, se observa que solo se necesitan dos escaneos de la base de datos de transacciones, *DB*: el primero reúne el conjunto de ítems frecuentes, el segundo construye el FP-tree.

2. Descubrir los patrones frecuentes utilizando FP-tree

Después de construir el árbol FP-tree se prosigue con el siguiente algoritmo para la minería de patrones frecuentes.

Algoritmo 2: Método FP-growth

*Entrada***:** FP-tree construido con base en el Algoritmo 1, usando DB y un umbral de soporte mínimo ξ.
*Salida***:** El conjunto completo de patrones frecuentes.
Método: Llamar FP-growth(FP-tree, null).

```
Procedimiento FP-growth (Tree, α)
{
    (1) Si Tree contiene un solo camino P
    (2) Entonces para cada  combinación (denotada como β) de los nodos en el camino
        P hacer
    (3)         generar el patrón β U α con soporte = soporte mínimo de nodos en β;
    (4) Sino para cada ai en la cabecera de Tree hacer  {
    (5)         generar el patrón β = ai U α con
      soporte = ai.soporte;
    (6) construir la base de patrones condicionales de β y luego el FP-tree condicional de
        β o Treeβ;
    (7) si Treeβ  ≠ Ø
    (8) entonces  llamar FP-growth (Treeβ, β) }
}
```

ANEXO D – IMPLEMENTACION DE LUCENE.NET

5. LUCENE.NET

Debido a las necesidades específicas del presente proyecto, de la implementación de Lucenet.NET, solo se utilizó la funcionalidad correspondiente al proceso de indexación, en sus fases de análisis léxico, eliminación de palabras vacías y stemming; no se necesitó reutilizar más fases del indexado, ni la funcionalidad de búsqueda, puesto que esta última se implementa como parte del desarrollo del proyecto. Las clases que se utilizaron en la indexación fueron: *IndexWriter*, *Directory*, *Analyzer*, *Document*, *Field* e *IndexReader*.

A continuación se presentan los pasos que se realizan en la fase de pre-procesamiento de documentos, en los cuales se reutiliza parte del código de Lucene.net:

1. **Crear el índice**: Se crea el índice, para lo cual se utilizan las clases *RAMDirectory* e *IndexWriter*. Se utiliza un directorio de tipo *RAMDirectory* porque no es necesario almacenar en disco los datos, ya que los temas de las consultas a la web por parte de los usuarios cambian mucho, por otra parte el tamaño de los datos no es tan elevado de modo que pueden ser cargados completamente en memoria RAM. Además, no era objetivo de esta investigación, realizar el indexado de documentos de la web, para ello se aprovecha el trabajo de buscadores tradicionales como Google.

2. **Indexar los documentos**: Se indexa el texto de cada documento en el índice que se creó:

    2.1. Para indexar un documento primero se crea un objeto de la clase Analyzer, el objeto *StopAnalyzer*, el cual crea palabras o *tokens* de los caracteres que encuentra adyacentes en el flujo de caracteres que recibió como documento, además cada caracter que encuentra lo convierte en minúscula, por otra parte los números que encuentre no los tiene en cuenta, es decir no los incluye en ningún *token*. Luego el *StopAnalyzer* elimina las palabras que se encuentran en la lista de *stopwords* que Lucene ya trae especificada, aunque se puede modificar si se requiere, en nuestro caso modificamos esta lista la cual se puede ver en detalle en el Anexo A.

    2.2. Después se crea un objeto *PorterStemFilter* que toma la raíz de cada palabra utilizando el algoritmo de stemming de Porter que viene implementado en Lucene, para ver en detalle el algoritmo remitirse al Anexo B.

    2.3. Se crea un objeto *Field* que corresponderá al contenido del documento (resumen), este campo tendrá las propiedades *Store*, *Index*, *TermVector*, es decir el campo se almacenará, se indexará y mantendrá un vector de los términos del documento con sus frecuencias asociadas.

    2.4. Se crea un objeto *Document* y a este se le adiciona el campo que se creó en el paso anterior.

    2.5. Finalmente al objeto *IndexWriter* se le adiciona el documento, es decir el documento queda guardado en el índice con todas las características que se le asignó.

3. **Consultar en el índice**: Se utiliza el *IndexReader* para acceder al índice, para ello se debe especificar el *RAMDirectory* para ubicar el índice creado. Una vez abierto el índice se utiliza el objeto *TermFreqVector* para obtener los términos de los documentos.

ANEXO E – PLANEACIÓN Y EJECUCIÓN DE PRUEBAS

6. PLANEACION DE PRUEBAS

6.1. PRUEBAS DE VALIDACIÓN

6.1.1. ALCANCE DE LA PRUEBA

Se pretende realizar dos tipos de pruebas a la aplicación web. Unas pruebas serán para evaluar la funcionalidad del sistema y las otras pruebas serán para evaluar la usabilidad del sistema. Las pruebas de funcionalidad se realizarán en dos sesiones cada uno de 45 minutos y la prueba de usabilidad se realizará en 20 minutos.

6.1.2. DURACIÓN ESTIMADA DE LA PRUEBA

- Introducción: 15 minutos
- Ejecución de la primera parte de la prueba: 45 minutos
- Descanso: 15 minutos
- Ejecución de la segunda parte de la prueba: 45 minutos
- Realizar test de usabilidad: 20 minutos
- Finalización de la prueba: 15 minutos

Tiempo total estimado: 2 horas y 35 minutos
Hora de inicio: 9:00 a.m.
Hora de fin: 11:35 a.m.

6.1.3. RECURSOS

Materiales

- Aplicación web instalada en el servidor SPAR
- Sala de computadores
- Computadores disponibles con conexión a internet (depende del número de participantes)
- Impresiones del test de usabilidad (depende del número de participantes)
- Impresiones de las consultas que se van a realizar (depende del número de participantes)

Recursos Humanos

- Participantes de la prueba (se espera contar con 40 estudiantes como mínimo)
- Orientadores de la prueba (2)

7. EJECUCIÓN DE PRUEBAS

7.1. PRUEBAS DE VALIDACIÓN

7.1.1. INTRODUCCION

Para esta evaluación, el meta buscador se encuentra implementado en una aplicación Web que puede ser accedida desde http://spar.unicauca.edu.co/gruweb. La evaluación consistió en realizar una serie de consultas y contestar unas preguntas para calificar la utilidad del meta buscador y aspectos relacionados con los resultados presentados. A continuación se presenta la introducción que se realizó a los usuarios antes de que iniciaran las pruebas sobre el meta buscador.

- *Explicar a los usuarios el objetivo de la prueba*: Se les dijo que se pretendía que los usuarios realizaran una serie de consultas y evaluaran los resultados del meta buscador.
- *Explicar a los usuarios el objetivo de la consulta*: Se presentó cada una de las preguntas del taller a realizar.
- *Explicar la aplicación web*: Se les explicó a los usuarios cómo ingresar la consulta y los parámetros a modificar en cada consulta, así mismo se les explico cómo realizar la calificación teniendo en cuenta los siguientes aspectos:

    - Los grupos se encuentran ordenados de forma descendente según su agrupamiento, es decir primero se encuentran los grupos más compactos o que contienen documentos que están más relacionados entre sí.
    - Los documentos se encuentran ordenados dentro del grupo según su pertenencia al grupo, es decir primero se encuentran los documentos más relacionados con el grupo.
    - Hay documentos que pueden estar en más de un grupo, porque se maneja solapamiento; se debería calificar cada documento sin importar que aparezca en más de un grupo.
    - Lo ideal es que se califiquen todos los documentos, pero debido a que la cantidad total de los mismos es alta, pueden calificar los documentos que deseen, en lo posible que sean unos documentos del inicio de la lista, unos documentos de la mitad y otros documentos del final de la lista.
    - Los documentos presentados provienen de consultas realizadas a Google, Yahoo y MSN.

7.1.2. DESARROLLO DE LA PRUEBA Y SEGUIMIENTO

Se solicitó la colaboración a 62 estudiantes del programa de Ingeniería de Sistemas para que realizaran las pruebas de la aplicación. La descripción de los grupos que participaron se presenta en la Tabla 1. La duración de cada prueba fue en promedio 1 hora.

Por otra parte, se estuvo pendiente del desarrollo de la prueba por parte de los usuarios, para aclarar dudas o resolver situaciones presentadas en cuanto a la consulta y los documentos recuperados. También se les orientó en la selección de los parámetros de la

consulta, los cuales debían variarse para determinar en el análisis de resultados cual combinación de parámetros arroja mejores resultados.

Grupo	N° Estudiantes	Asignatura	Profesor	Salón	N° de computadores	Fecha
1	6	Diferentes asignaturas	Ninguno	Oficina 105 - IPET	2	01/10/2009 02/10/2009
2	21	Gestión de Proyecto Informáticos	Ing. Luz Marina Sierra	Sala 1 - Sistemas	21	03/11/2009
3	21	Estructuras de Lenguaje	Ing. Jimena Timaná	Sala 4 – Sistemas	21	05/11/2009
4	14	Conceptos Avanzados de Bases de Datos	Ing. Martha Mendoza	Sala 4 – Sistemas	21	30/11/2009

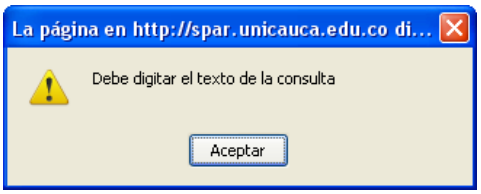
Tabla 1. Grupos participantes en las pruebas del meta buscador GruWeb

7.1.3. FINALIZACIÓN DE LA PRUEBA

Finalmente, se realizó un test de usabilidad, cuyos resultados se presentan en el Anexo E, además se preguntó a los usuarios sobre alguna opinión, sugerencia o algún aspecto adicional por mejorar en cuanto a la aplicación evaluada y se les dio las gracias por su participación.

7.2. PRUEBAS DE CAJA NEGRA

A continuación se presentan las pruebas de Caja Negra que se ejecutaron a la aplicación web. Estas pruebas se realizaron teniendo los casos de uso en los que se ingresan o modifican valores en las entradas del sistema, estos casos de uso son: Realizar Búsqueda (ver Tabla 2 y Tabla 3), Escoger Parámetros (ver Tabla 4 y Tabla 5) e Iniciar Sesión (ver Tabla 6 y Tabla 7). En los casos en los que fue útil se utilizó el Análisis de Valores Límite para desarrollar las pruebas de caja negra, por ejemplo en el caso de uso Escoger Parámetros.

Nombre campo	Valor entrada	Salida esperada	Salida presentada
Consulta a realizar	‘’	Indicación de que se debe ingresar el texto de la consulta	Mensaje de alerta mostrado: 
Consulta a	‘’	Indicación de	Mensaje de alerta mostrado:

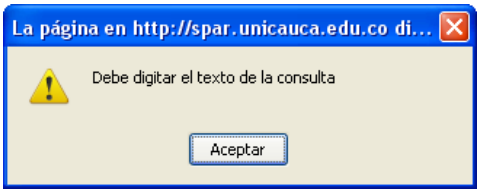
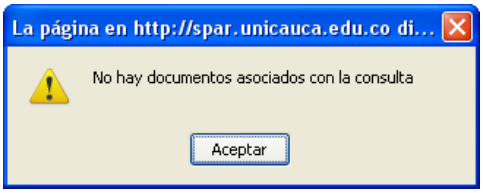
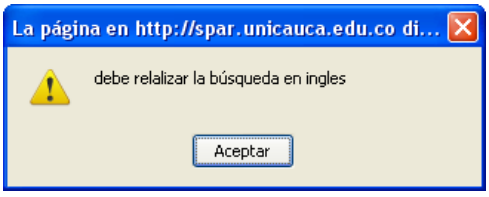
realizar		que se debe ingresar el texto de la consulta	
Consulta a realizar	'¿'	(Si con la consulta realizada no se obtienen documentos). Indicación de que se que no hay documentos asociados a la consulta	Mensaje de alerta mostrado: 
Consulta a realizar	'mineria de datos'	(Si la consulta no es en ingles). Indicación de que se debe realizar la consulta en ingles	Mensaje de alerta mostrado: 

Tabla 2. Entradas Inválidas para el Caso de Uso Realizar Búsqueda

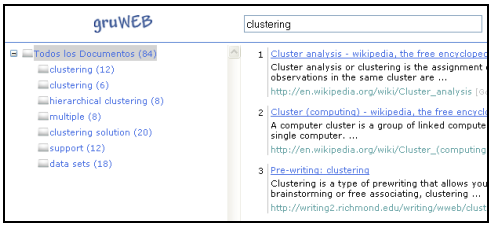
Nombre campo	Valor entrada	Salida esperada	Salida presentada
Consulta a realizar	'clustering'	Documentos asociados a la consulta presentados en grupos	Agrupamiento de los documentos asociados a la consulta: 

Tabla 3. Entradas Válidas para el Caso de Uso Realizar Búsqueda

Nombre campo	Valor entrada	Salida esperada	Salida presentada
Soporte mínimo [5-20]	'4'	No permita el ingreso del valor	Cambia el valor ingresado por el valor mínimo: 
Soporte mínimo [5-20]	'21'	No permita el ingreso del valor	Cambia el valor ingresado por el valor máximo: 
Tamaño de la Memoria de los Mejores Resultados (BMRS) [2-30]	'1'	No permita el ingreso del valor	Cambia el valor ingresado por el valor mínimo: 
Tamaño de la Memoria de los Mejores Resultados (BMRS) [2-30]	'31'	No permita el ingreso del valor	Cambia el valor ingresado por el valor máximo: 
Tamaño de la Memoria Armónica (HMS) [5-30]	'4'	No permita el ingreso del valor	Cambia el valor ingresado por el valor mínimo: 
Tamaño de la Memoria Armónica (HMS) [5-30]	'31'	No permita el ingreso del valor	Cambia el valor ingresado por el valor máximo: 
Tasa de Ajuste del Tono (PAR) [0-50]	'-1'	No permita el ingreso del valor	Cambia el valor ingresado por el valor mínimo: 
Tasa de Ajuste del Tono (PAR) [0-50]	'51'	No permita el ingreso del valor	Cambia el valor ingresado por el valor máximo: 
Tasa	'79'	No permita el	Cambia el valor ingresado por el

Considerada de la Memoria Armónica (HMCR) [80-100]		ingreso del valor	valor mínimo: 
Tasa Considerada de la Memoria Armónica (HMCR) [80-100]	‘101’	No permita el ingreso del valor	Cambia el valor ingresado por el valor máximo: 
Número de Improvisaciones (NI) [30-1000]	‘29’	No permita el ingreso del valor	Cambia el valor ingresado por el valor mínimo: 
Número de Improvisaciones (NI) [30-1000]	‘1001’	No permita el ingreso del valor	Cambia el valor ingresado por el valor máximo: 

Tabla 4. Entradas Inválidas para el Caso de Uso Escoger Parámetros

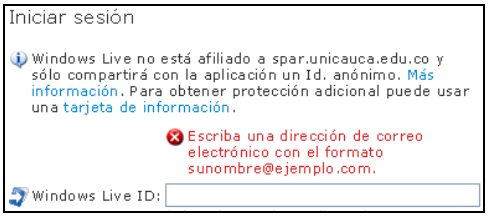
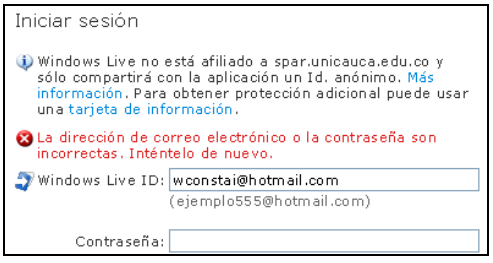
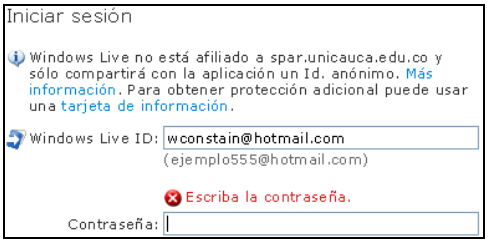
Nombre campo	Valor entrada	Salida esperada	Salida presentada
Soporte mínimo [5-20]	‘5’	Permita el ingreso del valor	Permite el valor ingresado: 
Soporte mínimo [5-20]	‘6’	Permita el ingreso del valor	Permite el valor ingresado: 
Soporte mínimo [5-20]	‘19’	Permita el ingreso del valor	Permite el valor ingresado: 
Soporte mínimo [5-20]	‘20’	Permita el ingreso del valor	Permite el valor ingresado: 
Tamaño de la Memoria	‘2’	Permita el ingreso del valor	Permite el valor ingresado:

de los Mejores Resultados (BMRS) [2-30]			
Tamaño de la Memoria de los Mejores Resultados (BMRS) [2-30]	'3'	Permita el ingreso del valor	Permite el valor ingresado: 
Tamaño de la Memoria de los Mejores Resultados (BMRS) [2-30]	'29'	Permita el ingreso del valor	Permite el valor ingresado: 
Tamaño de la Memoria de los Mejores Resultados (BMRS) [2-30]	'30'	Permita el ingreso del valor	Permite el valor ingresado: 
Tamaño de la Memoria Armónica (HMS) [5-30]	'5'	Permita el ingreso del valor	Permite el valor ingresado: 
Tamaño de la Memoria Armónica (HMS) [5-30]	'6'	Permita el ingreso del valor	Permite el valor ingresado: 
Tamaño de la Memoria Armónica (HMS) [5-30]	'29'	Permita el ingreso del valor	Permite el valor ingresado: 
Tamaño de la Memoria Armónica (HMS) [5-30]	'30'	Permita el ingreso del valor	Permite el valor ingresado: 
Tasa de Ajuste del	'0'	Permita el ingreso del valor	Permite el valor ingresado:

Tono (PAR) [0-50]			
Tasa de Ajuste del Tono (PAR) [0-50]	‘1’	Permita el ingreso del valor	Permite el valor ingresado: 
Tasa de Ajuste del Tono (PAR) [0-50]	‘49’	Permita el ingreso del valor	Permite el valor ingresado: 
Tasa de Ajuste del Tono (PAR) [0-50]	‘50’	Permita el ingreso del valor	Permite el valor ingresado: 
Tasa Considerada de la Memoria Armónica (HMCR) [80-100]	‘80’	Permita el ingreso del valor	Permite el valor ingresado: 
Tasa Considerada de la Memoria Armónica (HMCR) [80-100]	‘81’	Permita el ingreso del valor	Permite el valor ingresado: 
Tasa Considerada de la Memoria Armónica (HMCR) [80-100]	‘99’	Permita el ingreso del valor	Permite el valor ingresado: 
Tasa Considerada de la Memoria Armónica (HMCR) [80-100]	‘100’	Permita el ingreso del valor	Permite el valor ingresado: 
Número de Improvisaciones (NI) [30-1000]	‘30’	Permita el ingreso del valor	Permite el valor ingresado: 
Número de	‘31’	Permita el	Permite el valor ingresado:

Improvisaciones (NI) [30-1000]		ingreso del valor	
Número de Improvisaciones (NI) [30-1000]	'999'	Permita el ingreso del valor	Permite el valor ingresado: 
Número de Improvisaciones (NI) [30-1000]	'1000'	Permita el ingreso del valor	Permite el valor ingresado: 

Tabla 5. Entradas Válidas para el Caso de Uso Escoger Parámetros

Nombre campo	Valor entrada	Salida esperada	Salida presentada
Dirección de correo electrónico	''	Indicación de que debe ingresar un correo electrónico	Mensaje mostrado: 
Dirección de correo electrónico	'wconstai@hotmail.com'	Indicación de que se debe ingresar un correo valido	Mensaje mostrado: 
contraseña	''	Indicación de que debe ingresar la contraseña	Mensaje mostrado: 
contraseña	'*****'	Indicación de	Mensaje mostrado:

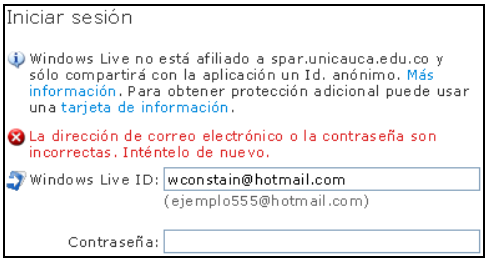
		que se debe ingresar una contraseña valida	

Tabla 6. Entradas Inválidas para el Caso de Uso Iniciar Sesión

Nombre campo	Valor entrada	Salida esperada	Salida presentada
Dirección de correo electrónico y contraseña valida	'wconstain@hotmail.com' y '**********'	Habilita la búsqueda y la sesión de usuario	Habilita la sesión de usuario y la búsqueda: 

Tabla 7. Entradas Válidas para el Caso de Uso Iniciar Sesión

ANEXO F – RESULTADOS DE LA ENCUESTA DE USABILIDAD

8. RESULTADOS DE LA ENCUESTA DE USABILIDAD

En este anexo se muestran los resultados obtenidos de la encuesta de usabilidad para el meta buscador GruWeb. A continuación se mostrará detalladamente cada pregunta con los resultados obtenidos en cada uno de los grupos de estudiantes en los que se realizaron las pruebas.

Para la realización de las pruebas a la aplicación se solicitó la colaboración a 62 estudiantes del programa de Ingeniería de Sistemas de la Universidad del Cauca. La descripción de los grupos de estudiantes que participaron se presenta en la Tabla 1Tabla 8. La duración de cada prueba fue en promedio 1 hora, en la cual al final de la prueba se realizó el test de usabilidad con una duración en promedio de 15 minutos.

Grupo	N° Estudiantes	Asignatura	Profesor	Salón	N° de computadores	Fecha
1	6	Diferentes asignaturas	Ninguno	Oficina 105 - IPET	2	01/10/2009 02/10/2009
2	21	Gestión de Proyecto Informáticos (GPI)	Ing. Luz Marina Sierra	Sala 1 - Sistemas	21	03/11/2009
3	21	Estructuras de Lenguaje	Ing. Jimena Timaná	Sala 4 – Sistemas	21	05/11/2009
4	14	Conceptos Avanzados de Bases de Datos (CABD)	Ing. Martha Mendoza	Sala 4 – Sistemas	21	30/11/2009

Tabla 8. Grupos participantes en las pruebas del meta buscador GruWeb

Resultados obtenidos en las preguntas del test de usabilidad en el Grupo 2 (Estudiantes de Gestión de Proyectos Informáticos)

Ver Figura 1 a Figura 14.

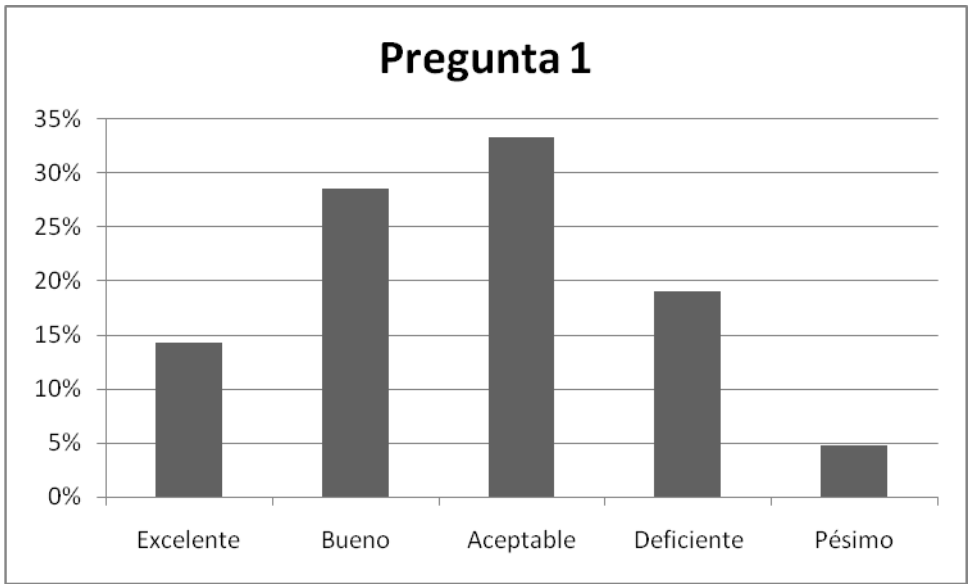


Figura 1. Resultados en la Pregunta 1 de la Encuesta de Usabilidad en el Grupo 2 de Usuarios

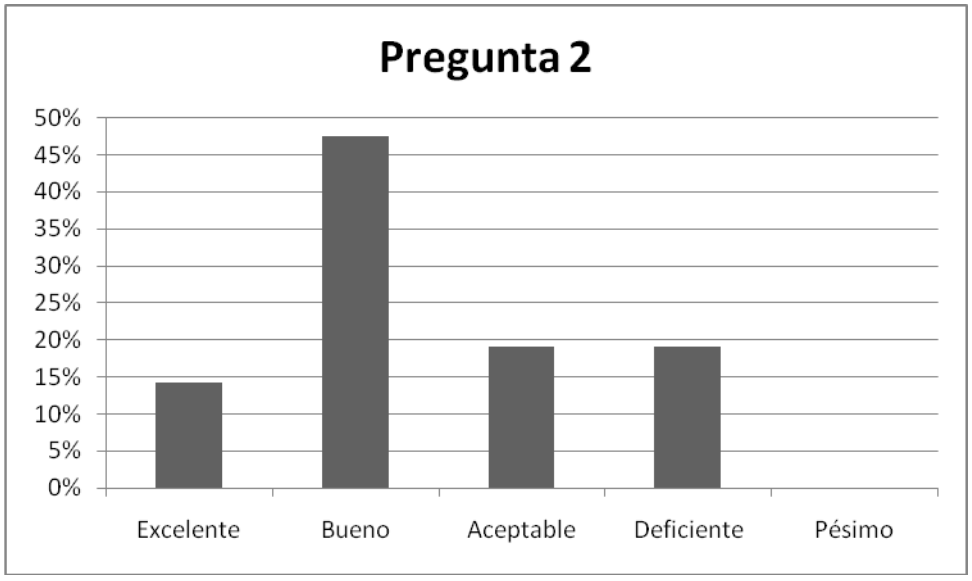


Figura 2. Resultados en la Pregunta 2 de la Encuesta de Usabilidad en el Grupo 2 de Usuarios

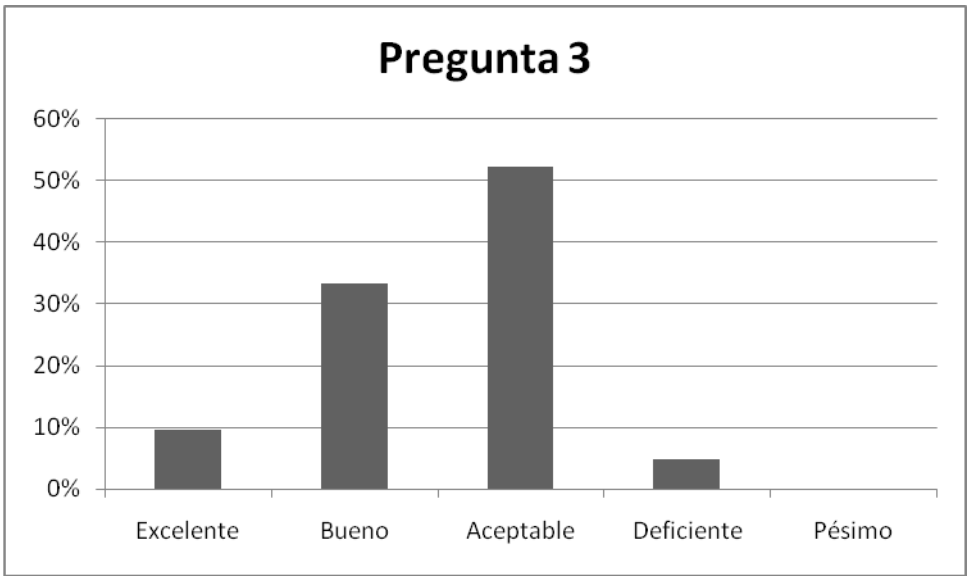


Figura 3. Resultados en la Pregunta 3 de la Encuesta de Usabilidad en el Grupo 2 de Usuarios

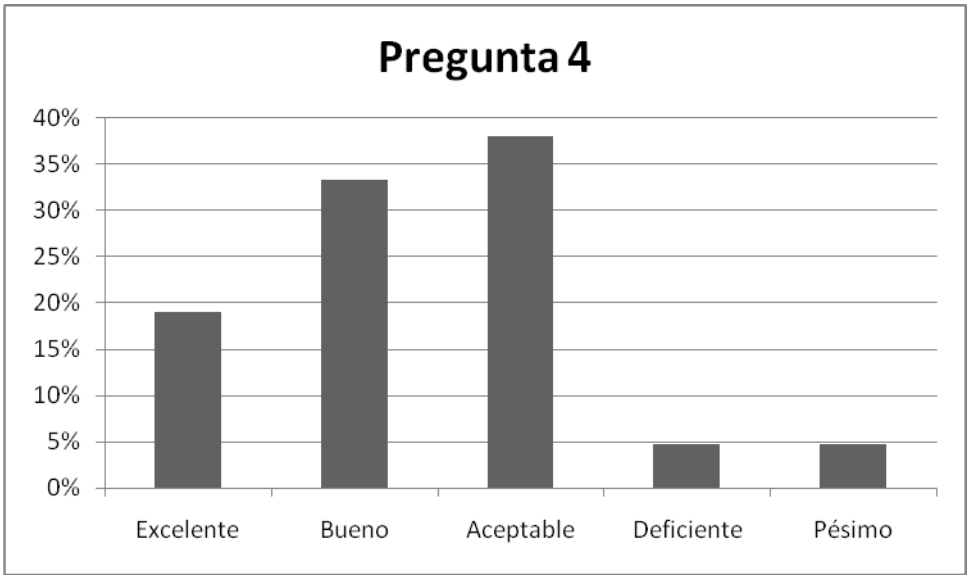


Figura 4. Resultados en la Pregunta 4 de la Encuesta de Usabilidad en el Grupo 2 de Usuarios

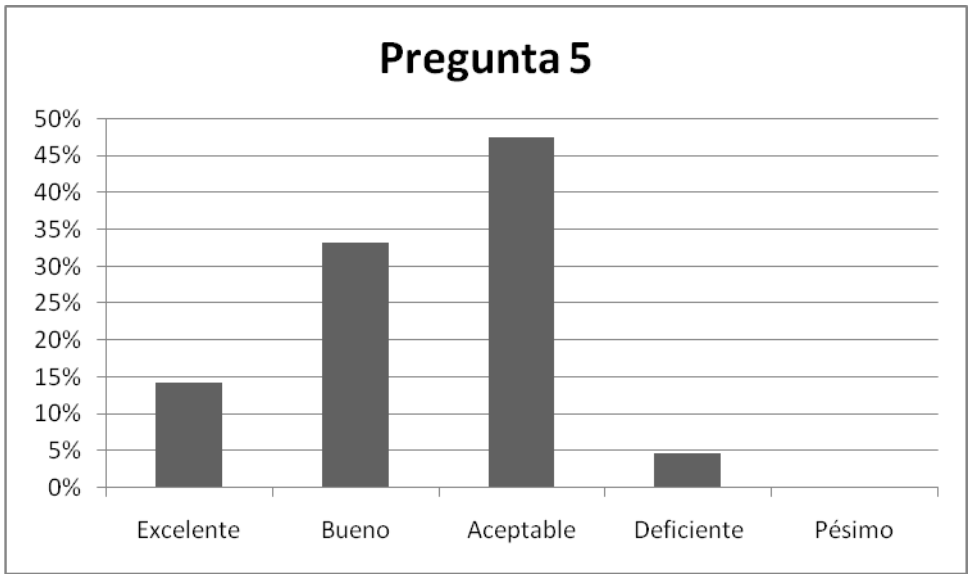


Figura 5. Resultados en la Pregunta 5 de la Encuesta de Usabilidad en el Grupo 2 de Usuarios

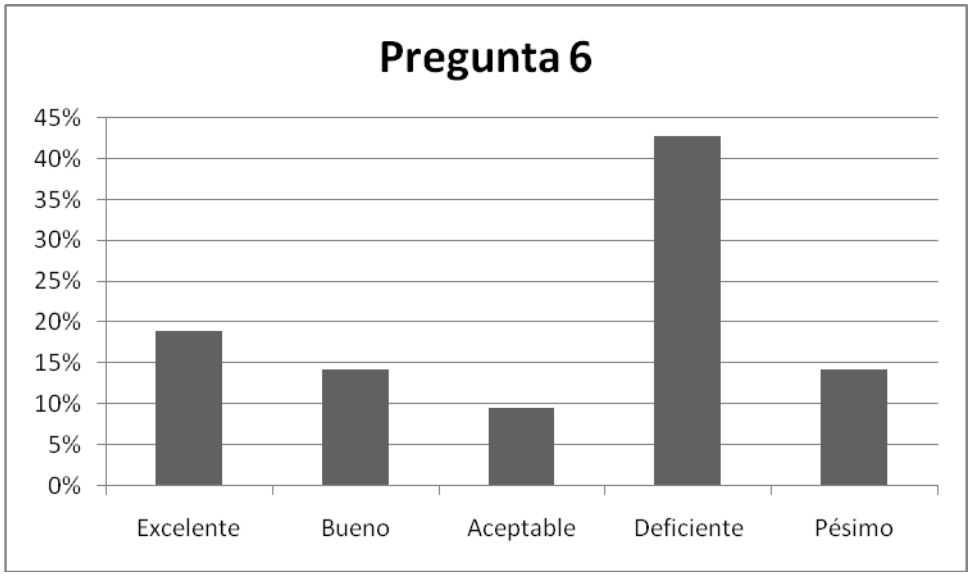


Figura 6. Resultados en la Pregunta 6 de la Encuesta de Usabilidad en el Grupo 2 de Usuarios

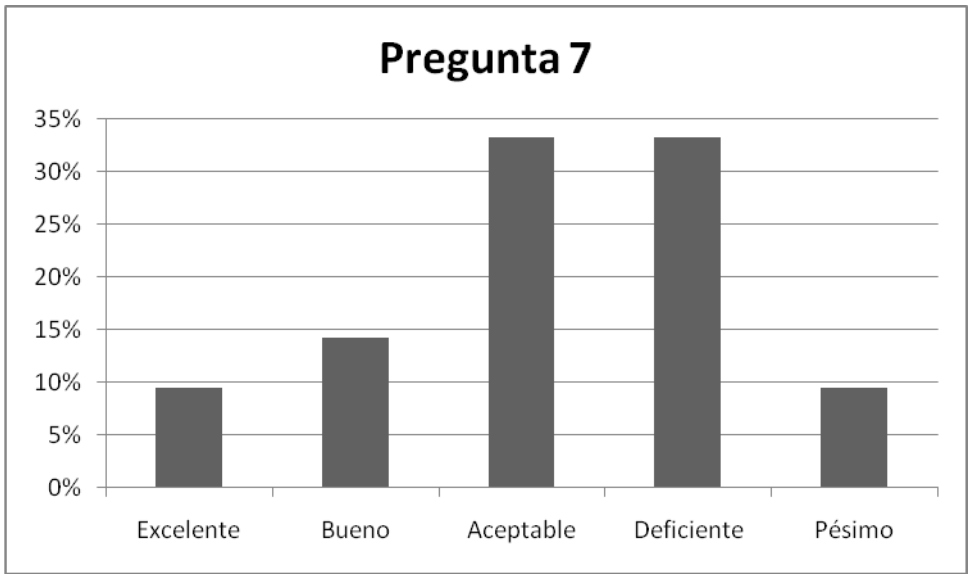


Figura 7. Resultados en la Pregunta 7 de la Encuesta de Usabilidad en el Grupo 2 de Usuarios

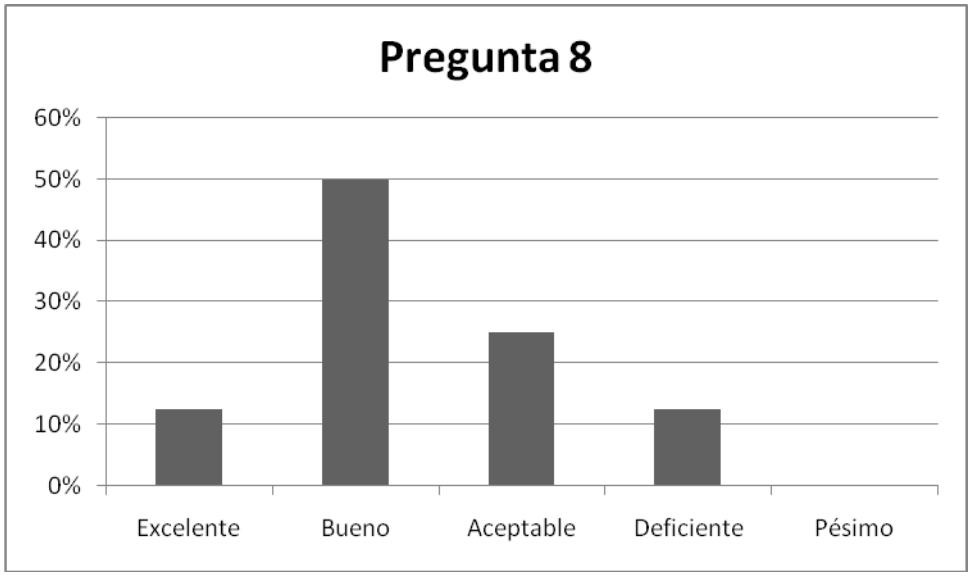


Figura 8. Resultados en la Pregunta 8 de la Encuesta de Usabilidad en el Grupo 2 de Usuarios

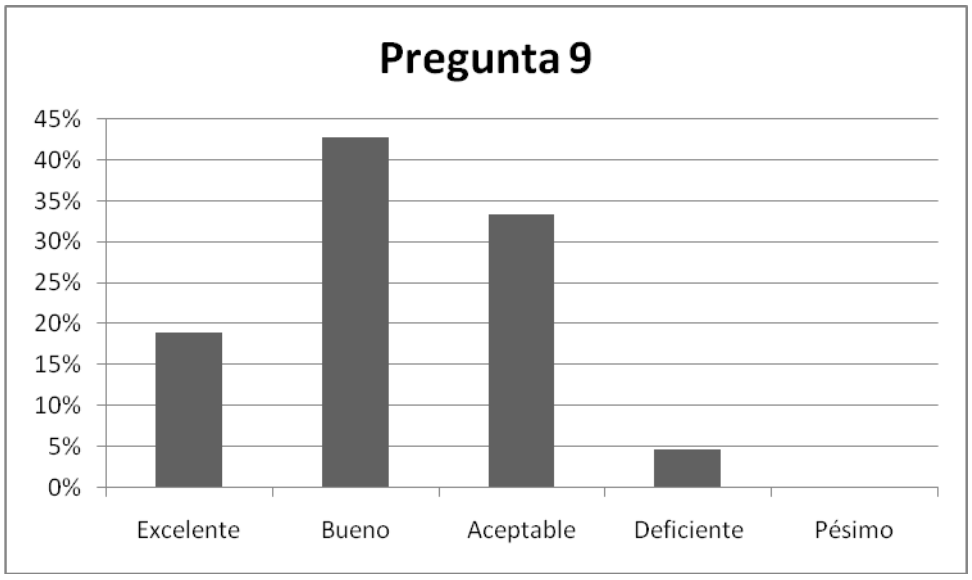


Figura 9. Resultados en la Pregunta 9 de la Encuesta de Usabilidad en el Grupo 2 de Usuarios

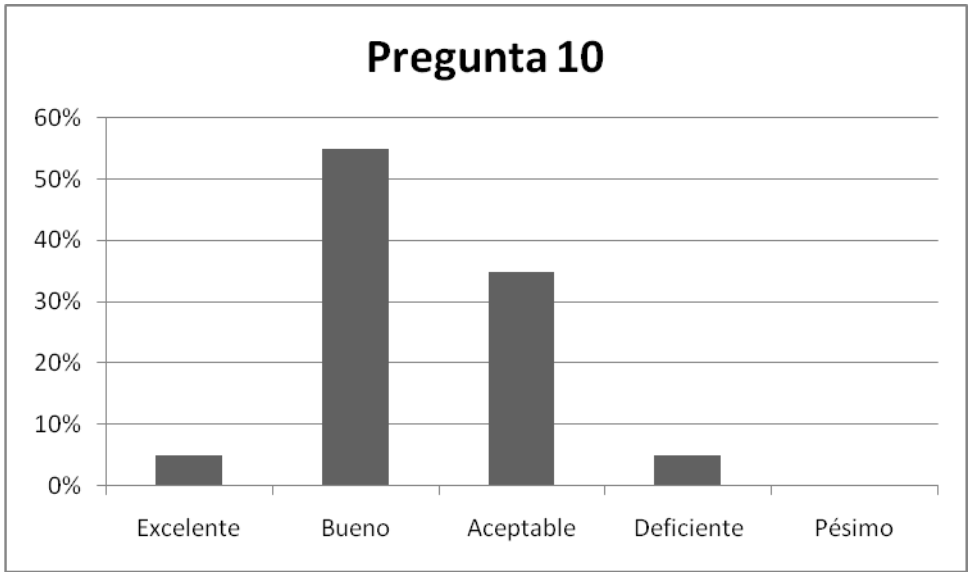


Figura 10. Resultados en la Pregunta 10 de la Encuesta de Usabilidad en el Grupo 2 de Usuarios

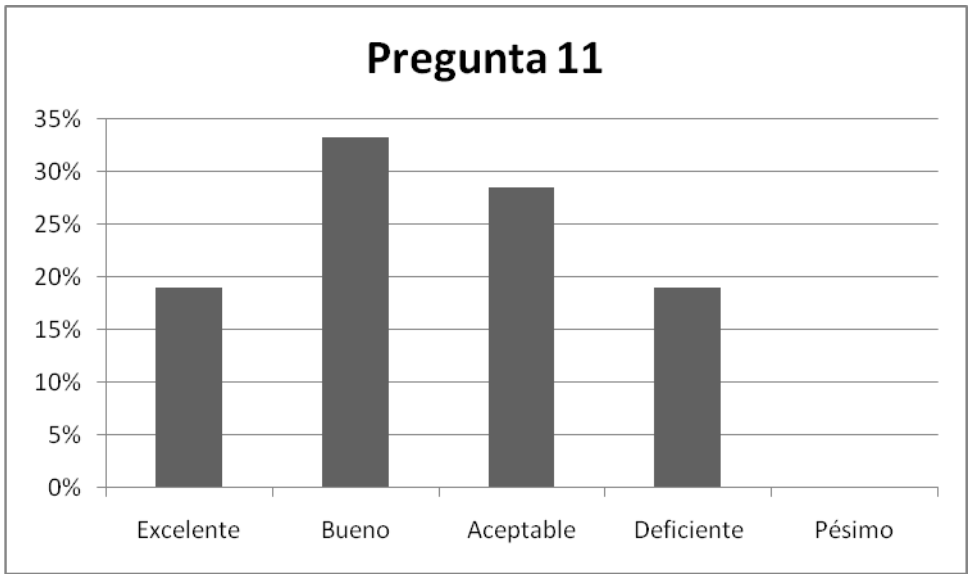


Figura 11. Resultados en la Pregunta 11 de la Encuesta de Usabilidad en el Grupo 2 de Usuarios

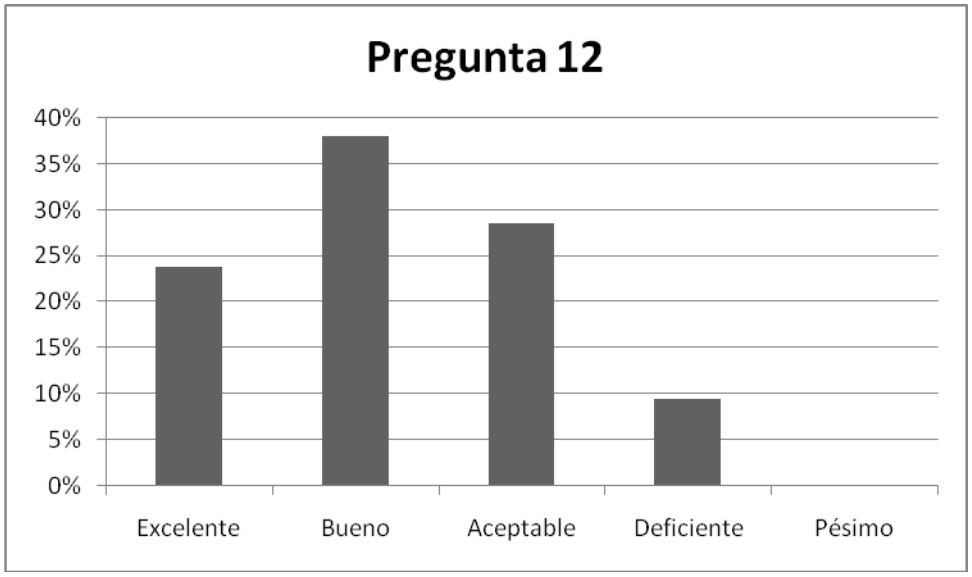


Figura 12. Resultados en la Pregunta 12 de la Encuesta de Usabilidad en el Grupo 2 de Usuarios

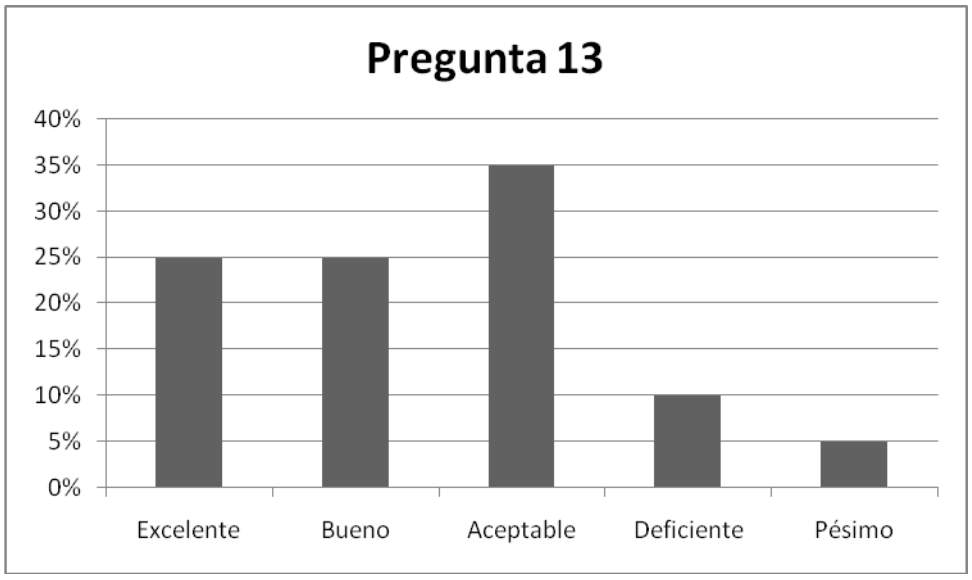


Figura 13. Resultados en la Pregunta 13 de la Encuesta de Usabilidad en el Grupo 2 de Usuarios

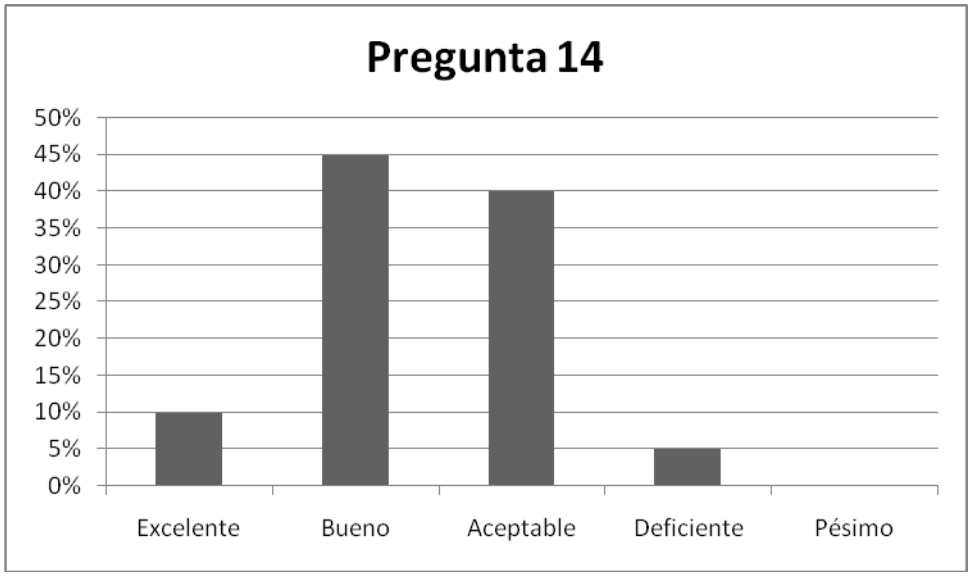


Figura 14. Resultados en la Pregunta 14 de la Encuesta de Usabilidad en el Grupo 2 de Usuarios

Resultados obtenidos en las preguntas del test de usabilidad en el Grupo 3 (Estudiantes de Estructuras de Lenguaje)

Ver Figura 15 a Figura 28.

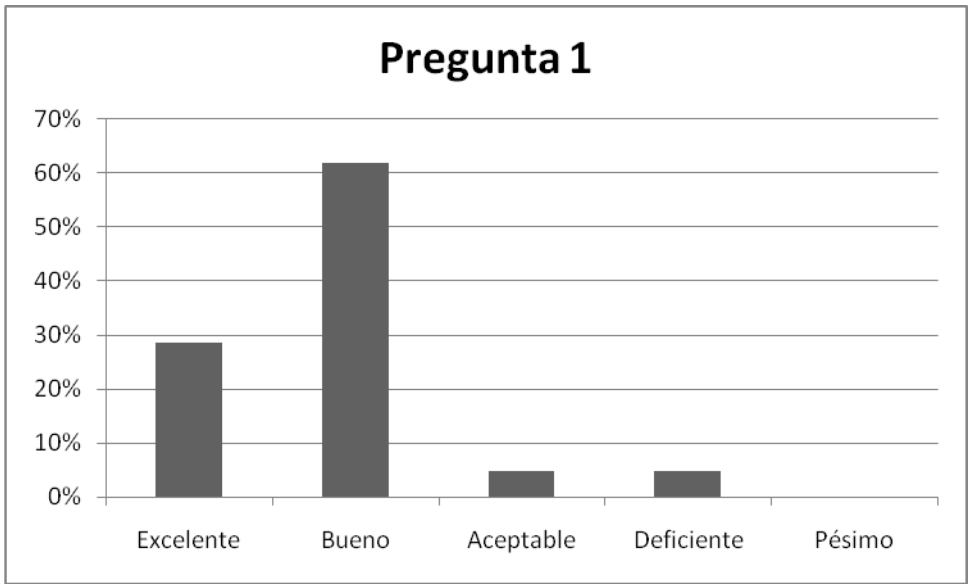


Figura 15. Resultados en la Pregunta 1 de la Encuesta de Usabilidad en el Grupo 3 de Usuarios

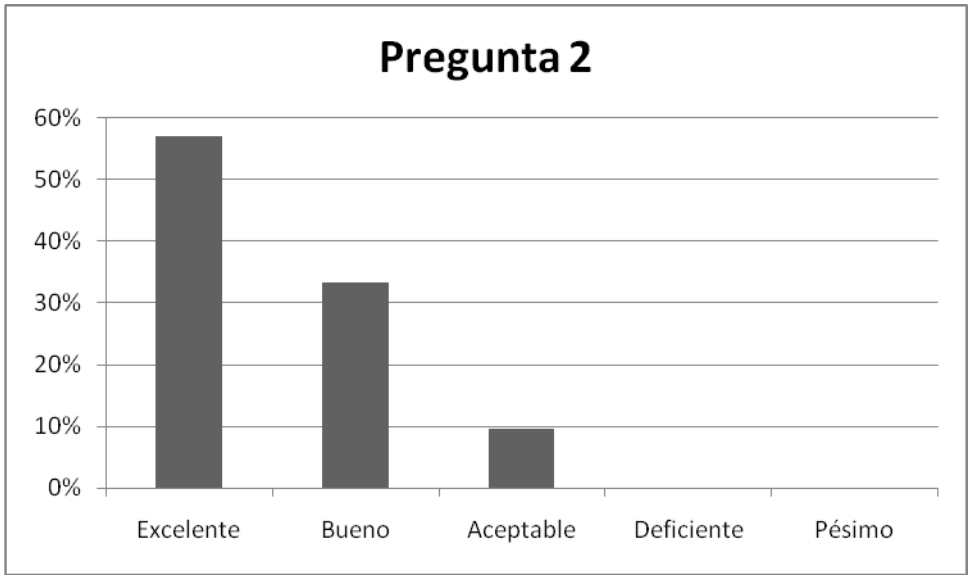


Figura 16. Resultados en la Pregunta 2 de la Encuesta de Usabilidad en el Grupo 3 de Usuarios

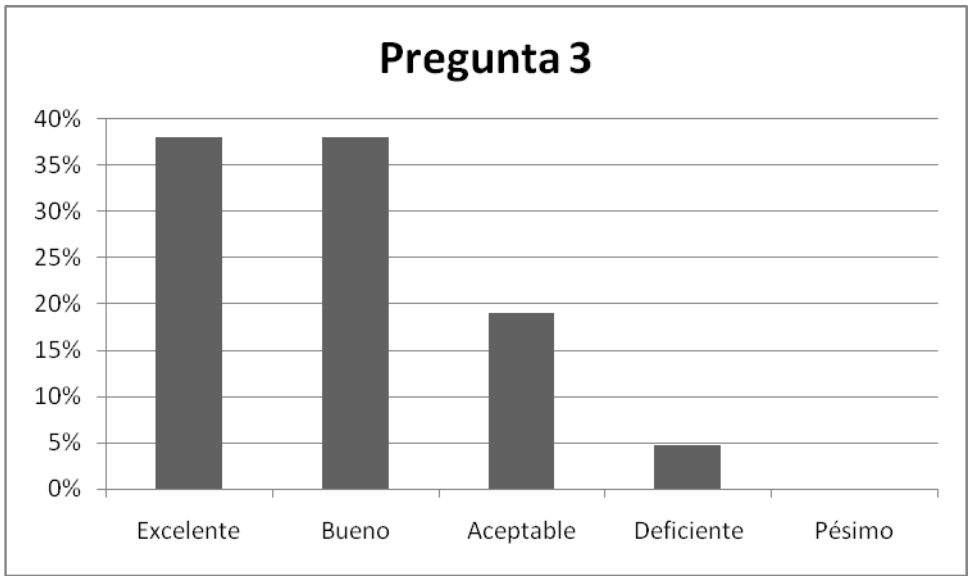


Figura 17. Resultados en la Pregunta 3 de la Encuesta de Usabilidad en el Grupo 3 de Usuarios

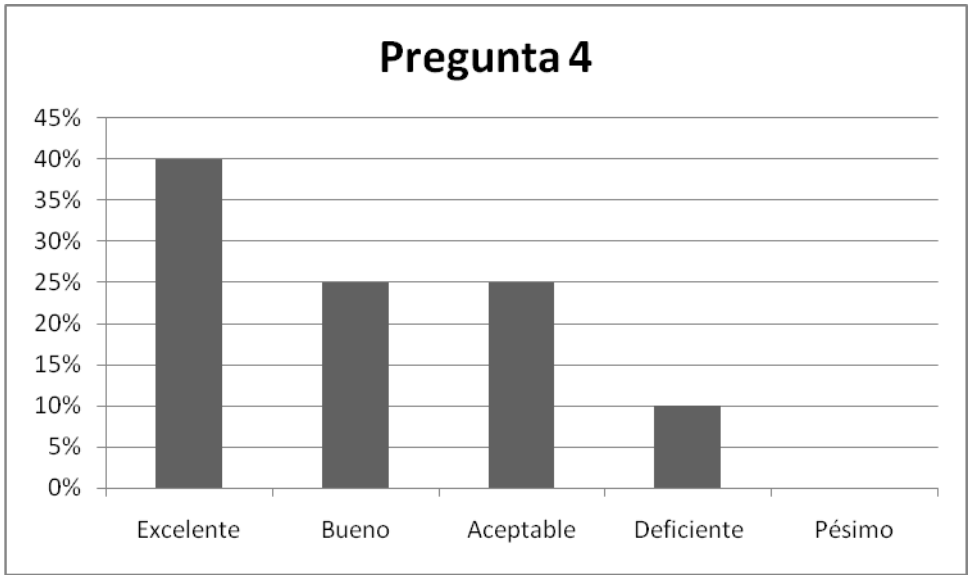


Figura 18. Resultados en la Pregunta 4 de la Encuesta de Usabilidad en el Grupo 3 de Usuarios

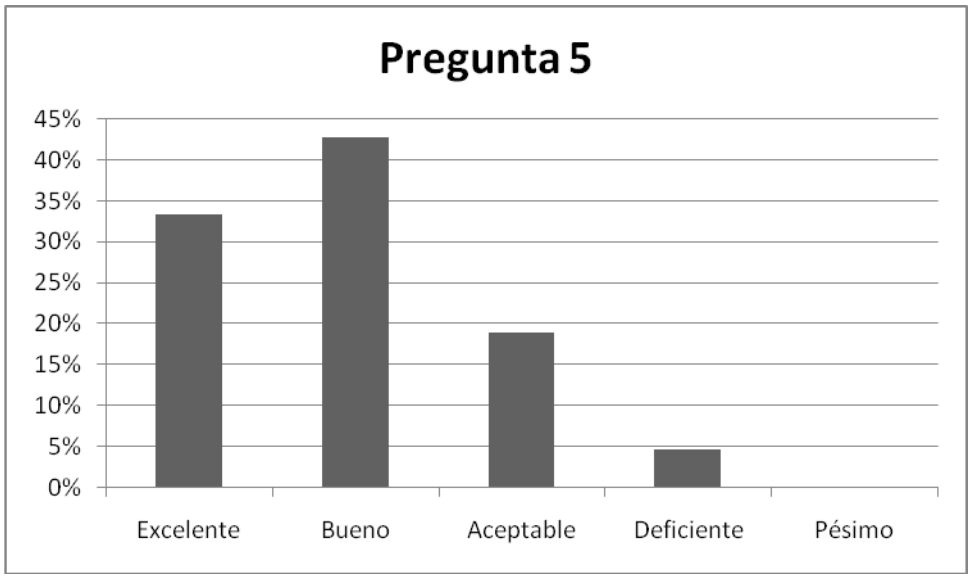


Figura 19. Resultados en la Pregunta 5 de la Encuesta de Usabilidad en el Grupo 3 de Usuarios

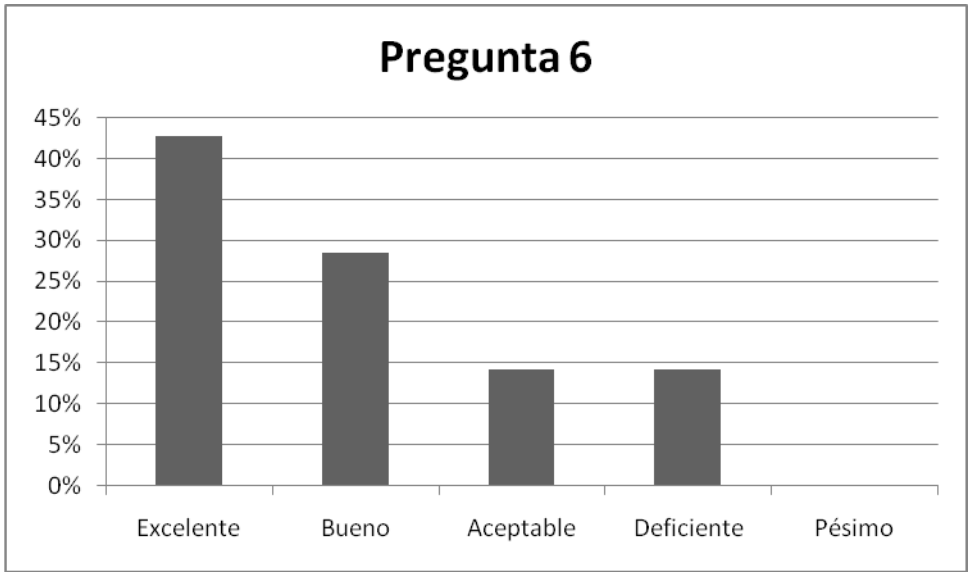


Figura 20. Resultados en la Pregunta 6 de la Encuesta de Usabilidad en el Grupo 3 de Usuarios

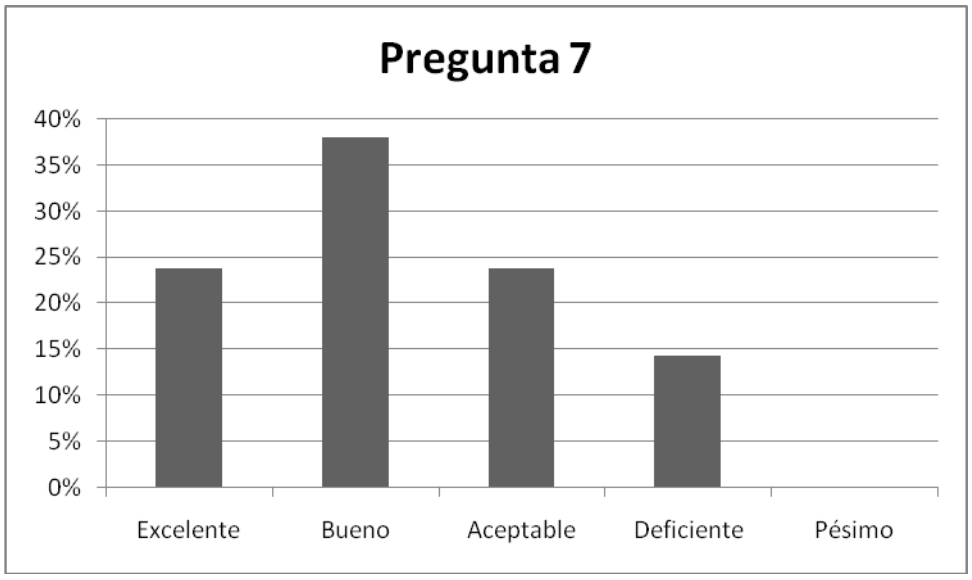


Figura 21. Resultados en la Pregunta 7 de la Encuesta de Usabilidad en el Grupo 3 de Usuarios

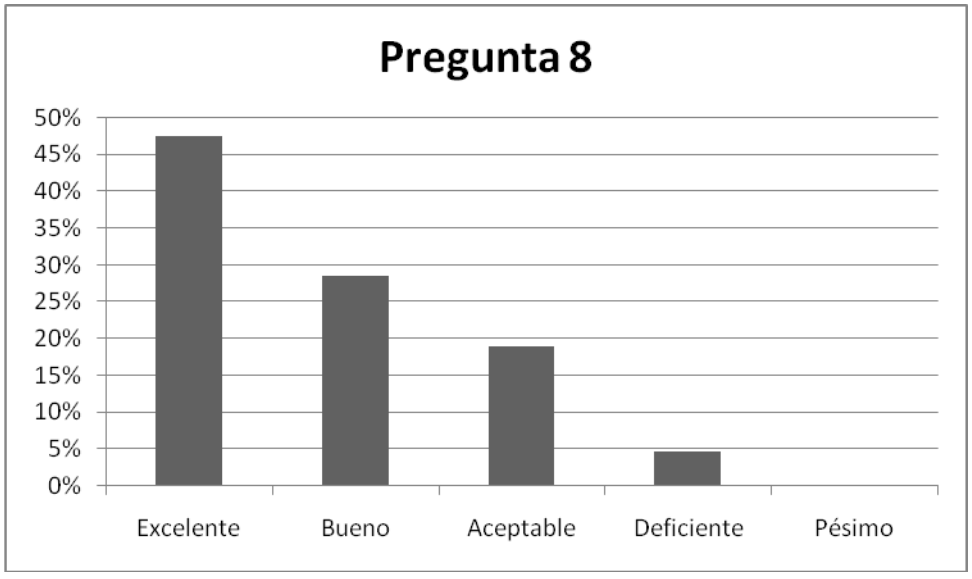


Figura 22. Resultados en la Pregunta 8 de la Encuesta de Usabilidad en el Grupo 3 de Usuarios

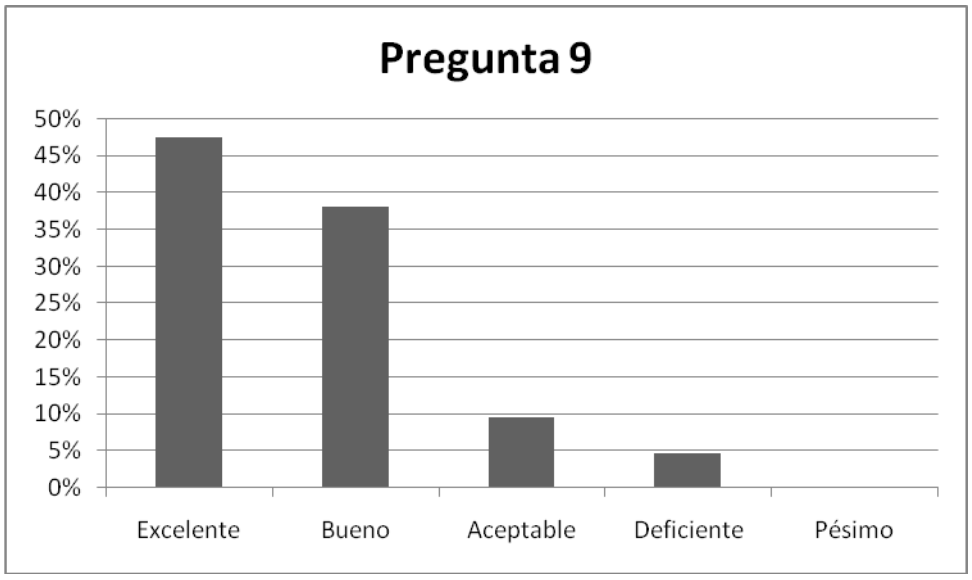


Figura 23. Resultados en la Pregunta 9 de la Encuesta de Usabilidad en el Grupo 3 de Usuarios

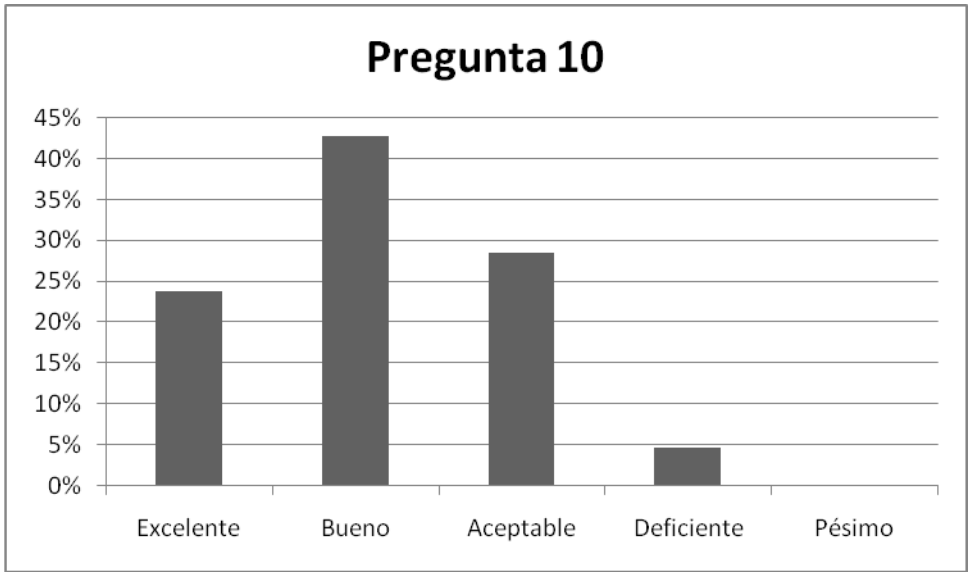


Figura 24. Resultados en la Pregunta 10 de la Encuesta de Usabilidad en el Grupo 3 de Usuarios

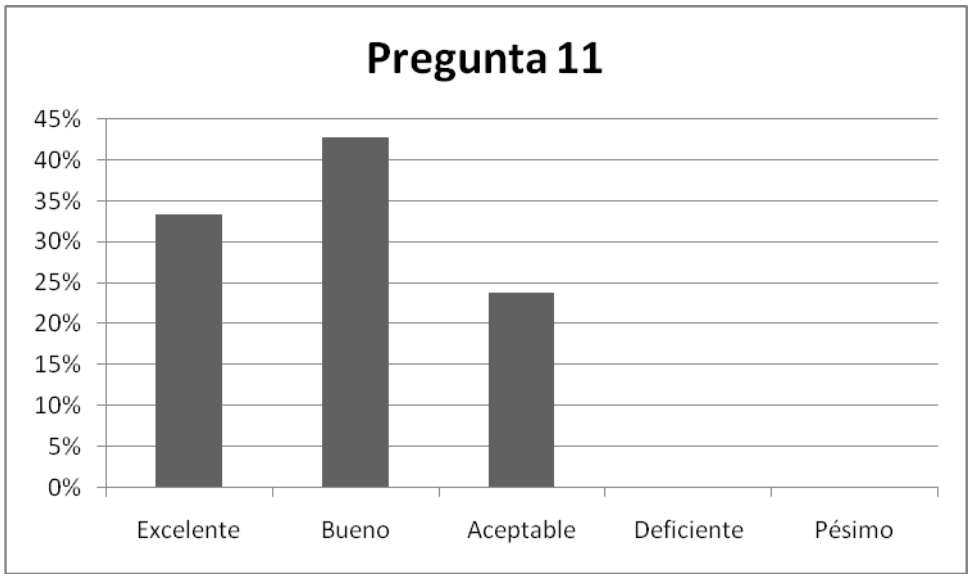


Figura 25. Resultados en la Pregunta 11 de la Encuesta de Usabilidad en el Grupo 3 de Usuarios

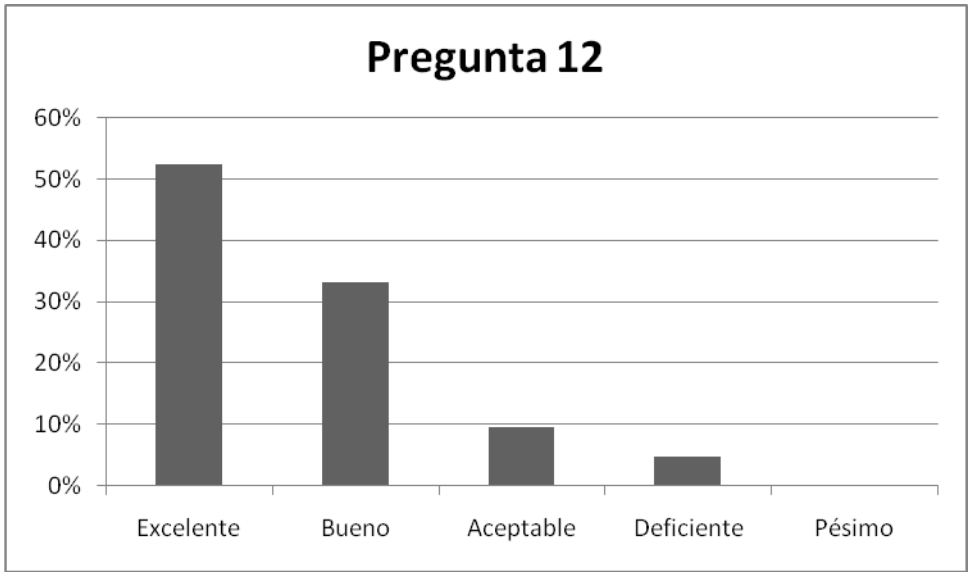


Figura 26. Resultados en la Pregunta 12 de la Encuesta de Usabilidad en el Grupo 3 de Usuarios

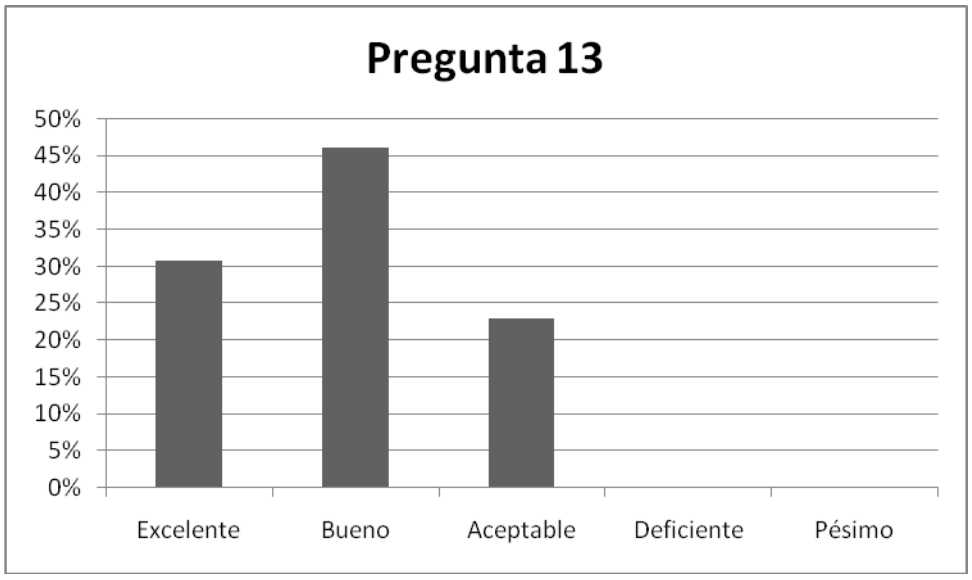


Figura 27. Resultados en la Pregunta 13 de la Encuesta de Usabilidad en el Grupo 3 de Usuarios

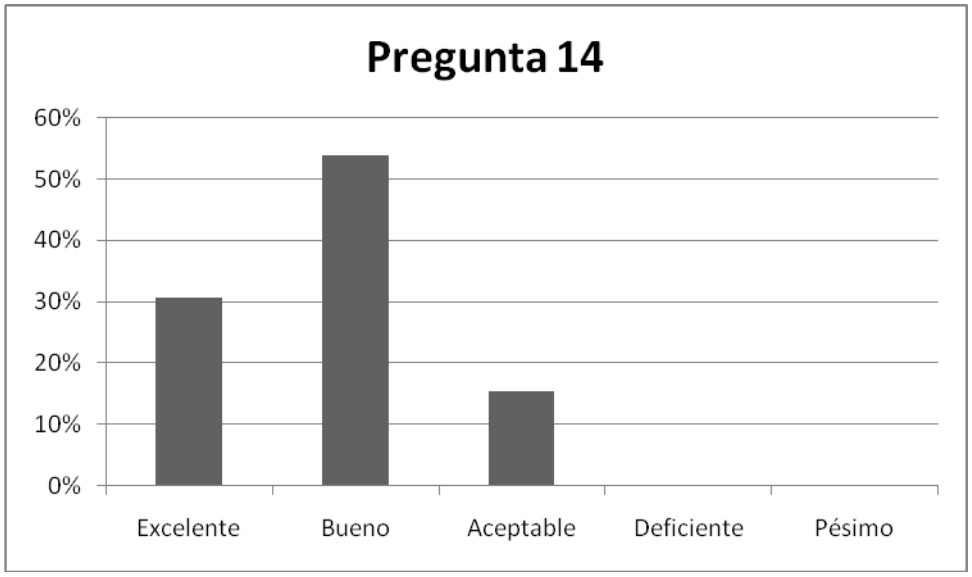


Figura 28. Resultados en la Pregunta 14 de la Encuesta de Usabilidad en el Grupo 3 de Usuarios

Resultados obtenidos en las preguntas del test de usabilidad en el Grupo 4 (Estudiantes de Conceptos Avanzados de Bases de Datos)

Ver Figura 29 a Figura 42.

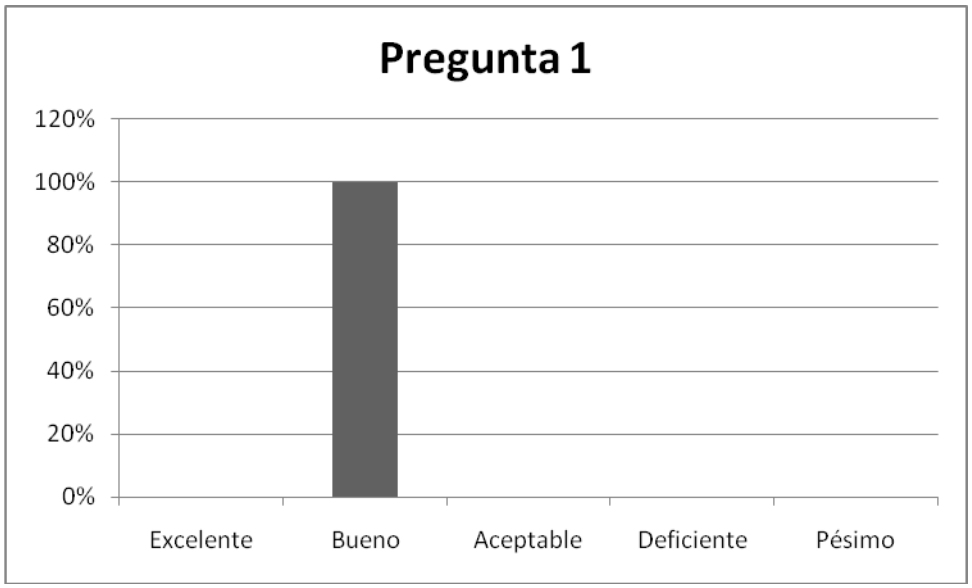


Figura 29. Resultados en la Pregunta 1 de la Encuesta de Usabilidad en el Grupo 4 de Usuarios

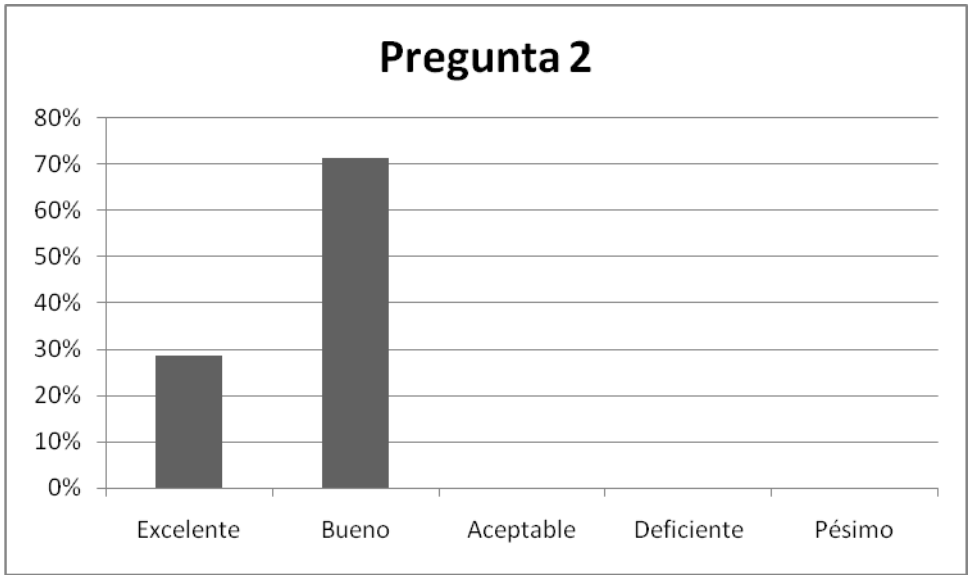


Figura 30. Resultados en la Pregunta 2 de la Encuesta de Usabilidad en el Grupo 4 de Usuarios

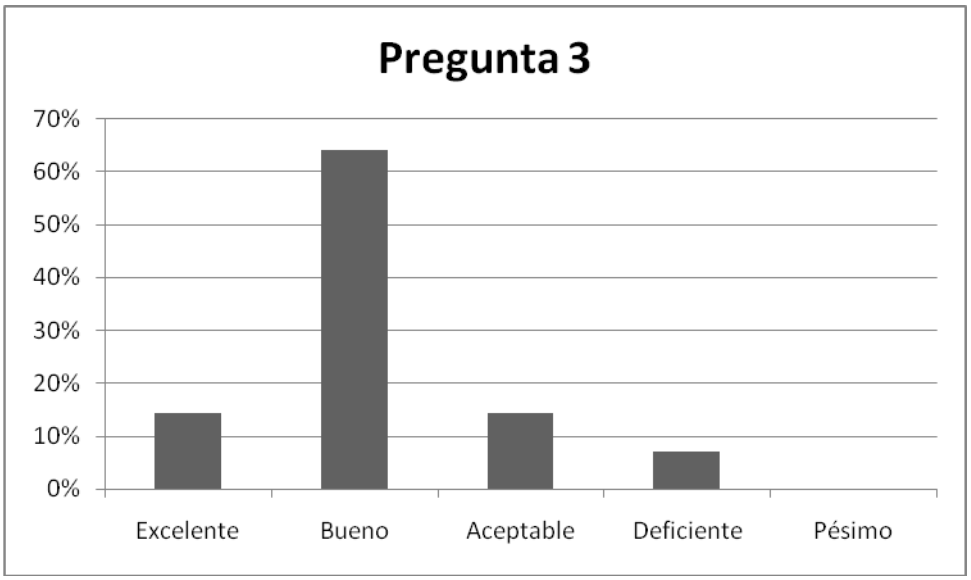


Figura 31. Resultados en la Pregunta 3 de la Encuesta de Usabilidad en el Grupo 4 de Usuarios

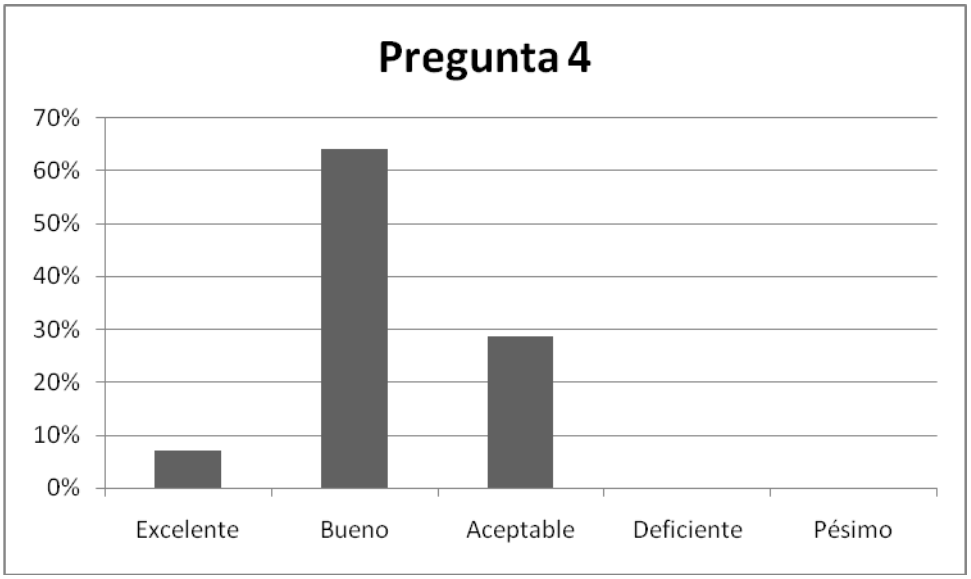


Figura 32. Resultados en la Pregunta 4 de la Encuesta de Usabilidad en el Grupo 4 de Usuarios

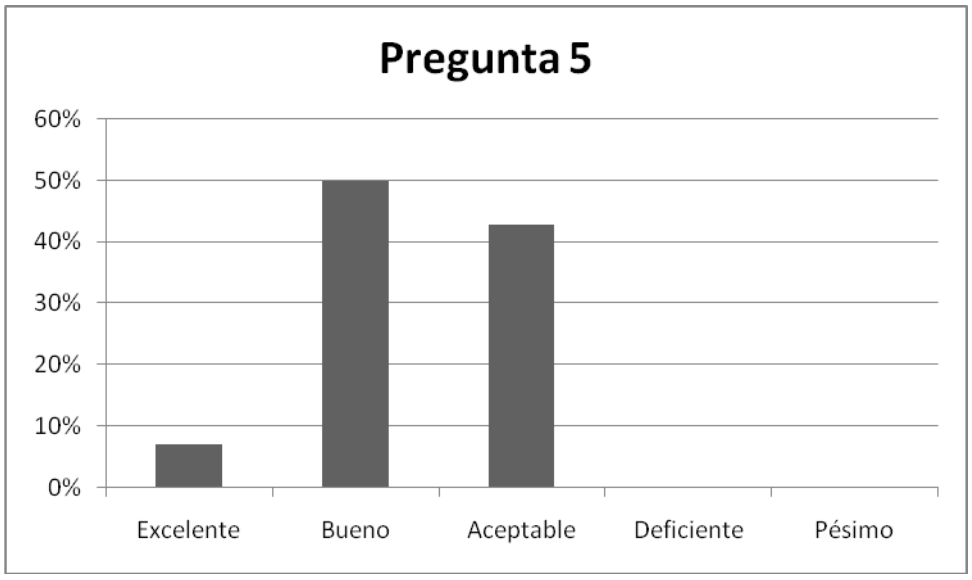


Figura 33. Resultados en la Pregunta 5 de la Encuesta de Usabilidad en el Grupo 4 de Usuarios

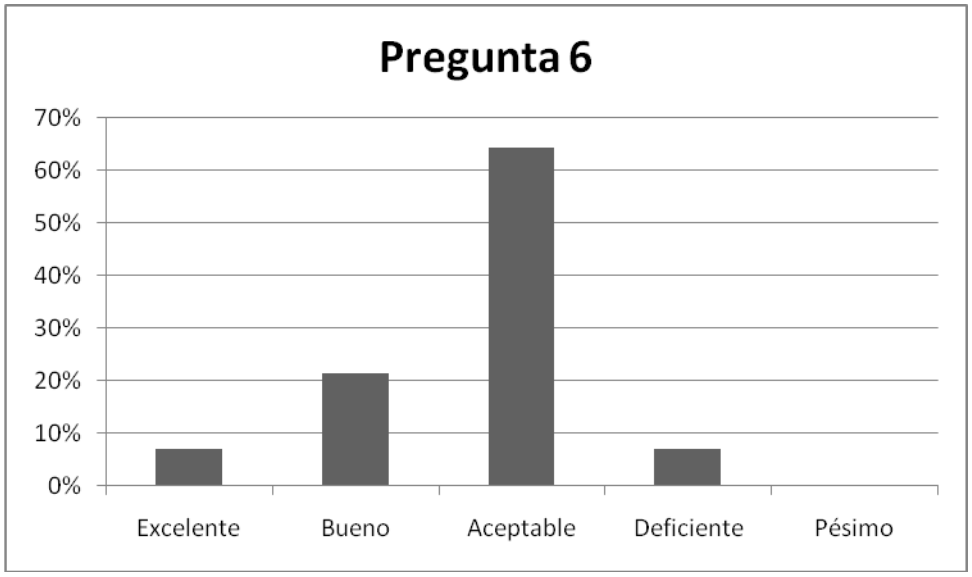


Figura 34. Resultados en la Pregunta 6 de la Encuesta de Usabilidad en el Grupo 4 de Usuarios

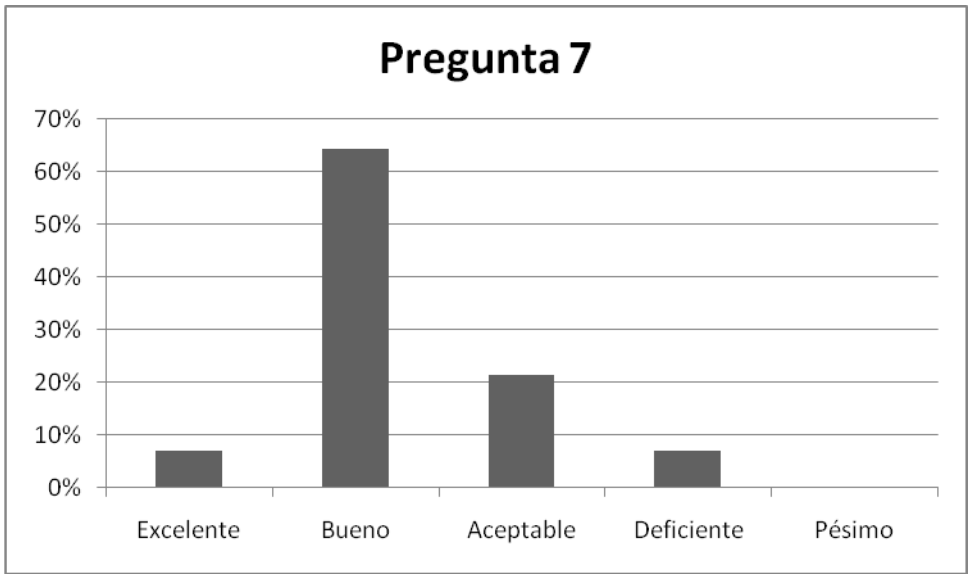


Figura 35. Resultados en la Pregunta 7 de la Encuesta de Usabilidad en el Grupo 4 de Usuarios

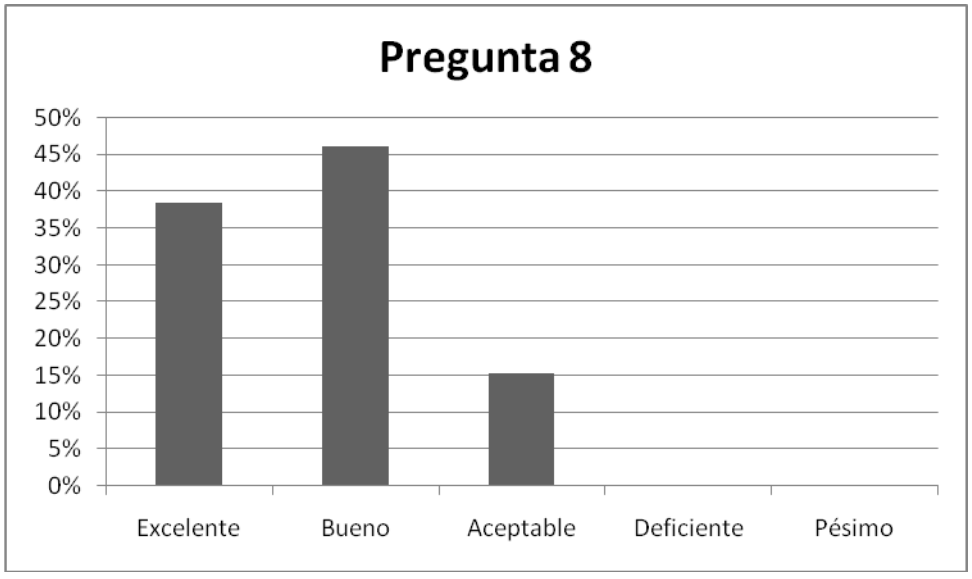


Figura 36. Resultados en la Pregunta 8 de la Encuesta de Usabilidad en el Grupo 4 de Usuarios

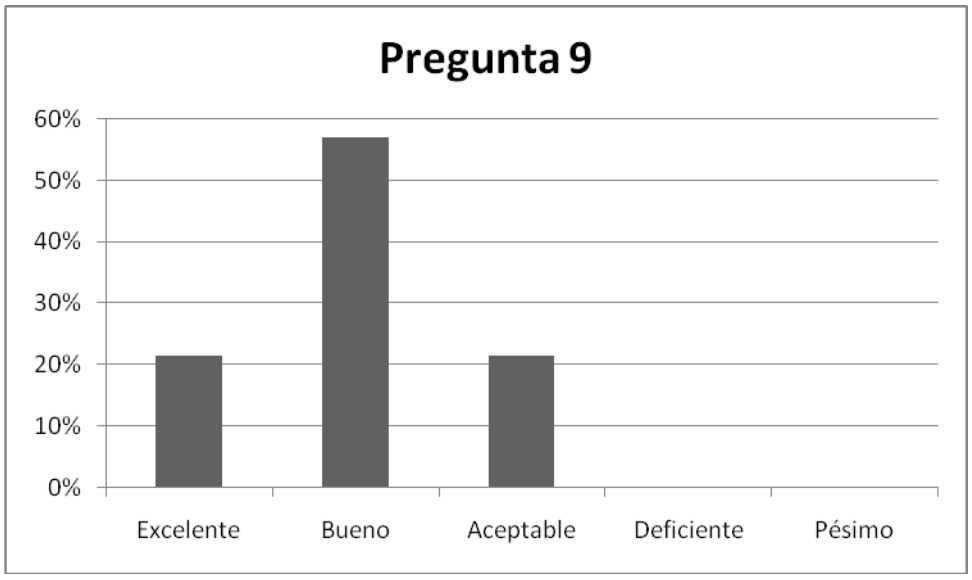


Figura 37. Resultados en la Pregunta 9 de la Encuesta de Usabilidad en el Grupo 4 de Usuarios

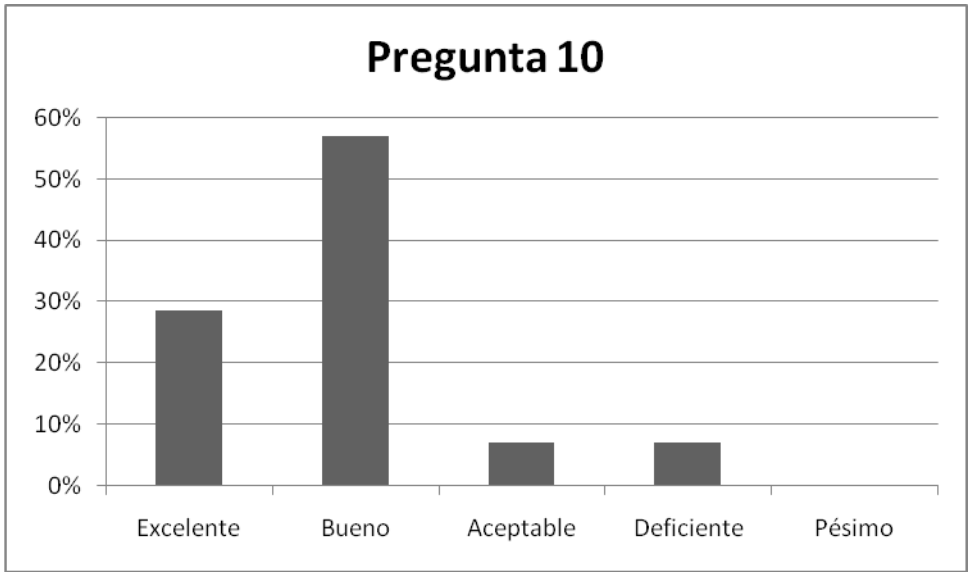


Figura 38. Resultados en la Pregunta 10 de la Encuesta de Usabilidad en el Grupo 4 de Usuarios

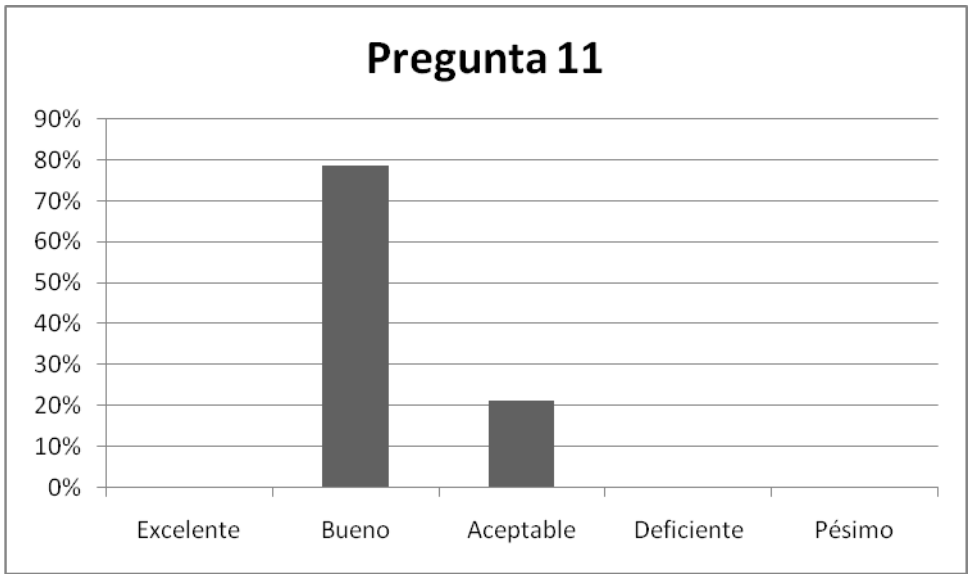


Figura 39. Resultados en la Pregunta 11 de la Encuesta de Usabilidad en el Grupo 4 de Usuarios

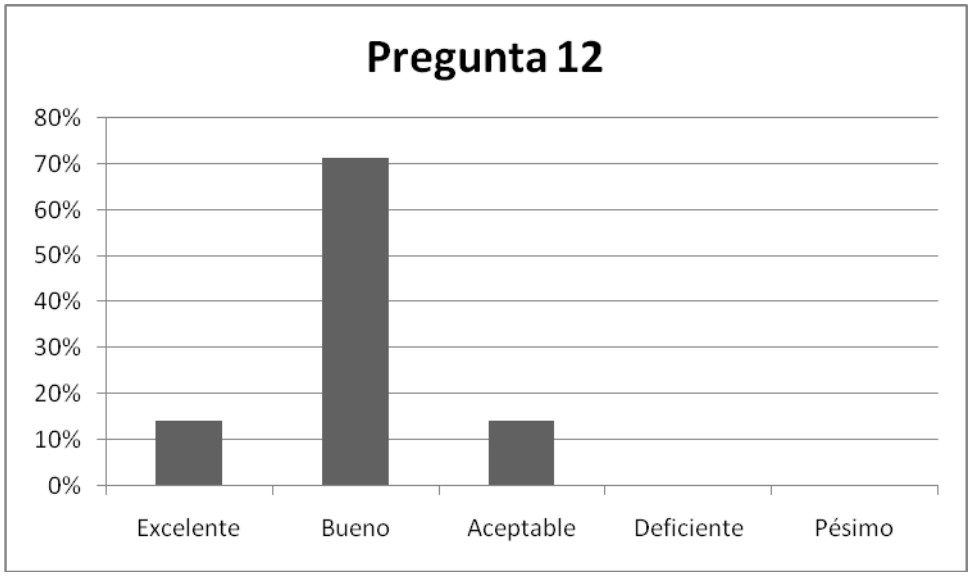


Figura 40. Resultados en la Pregunta 12 de la Encuesta de Usabilidad en el Grupo 4 de Usuarios

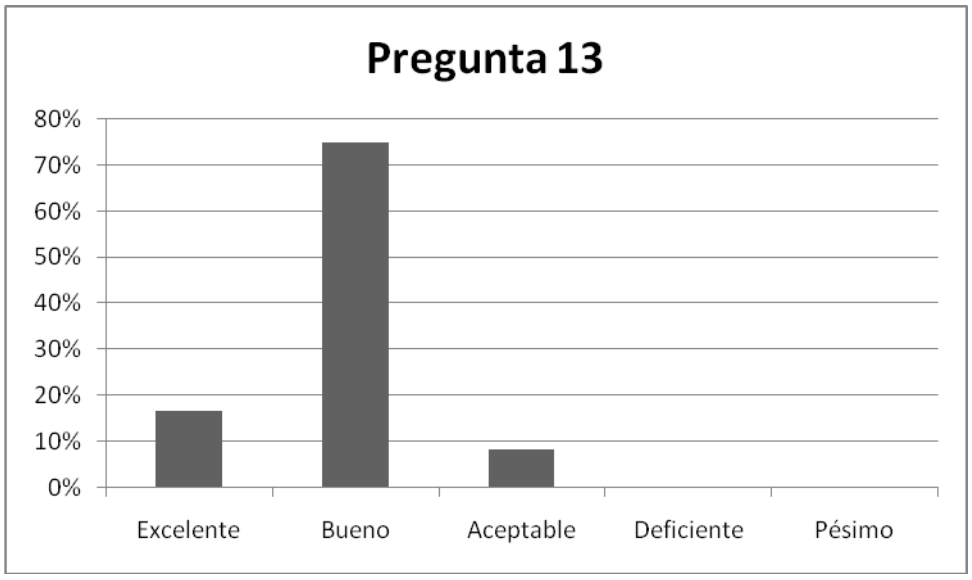


Figura 41. Resultados en la Pregunta 13 de la Encuesta de Usabilidad en el Grupo 4 de Usuarios

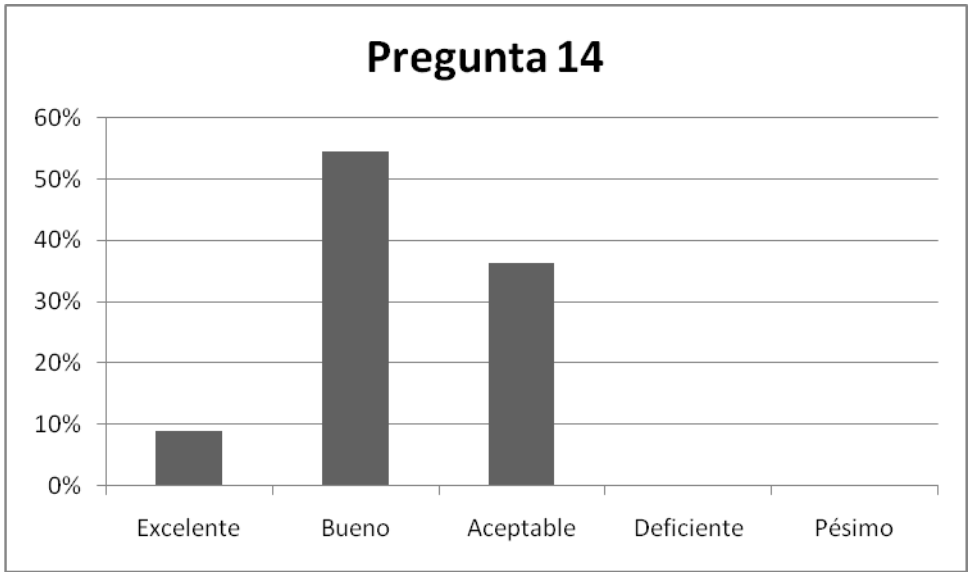


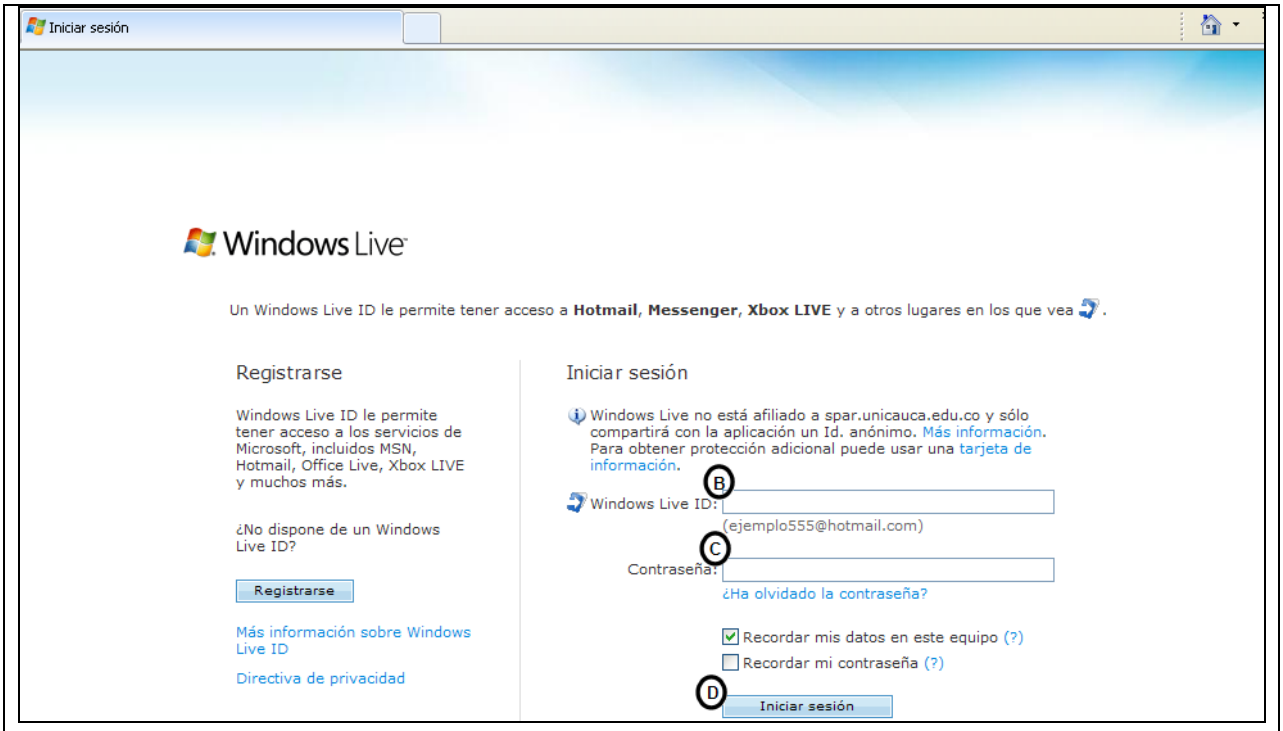
Figura 42. Resultados en la Pregunta 14 de la Encuesta de Usabilidad en el Grupo 4 de Usuarios

ANEXO G – CASOS DE USO REALES

9. CASOS DE USO REALES

A continuación se presentan los casos de uso reales Iniciar Sesión (ver Tabla 9), Escoger Parámetros (ver Tabla 10) y Calificar Grupos y Documentos (ver Tabla 11).

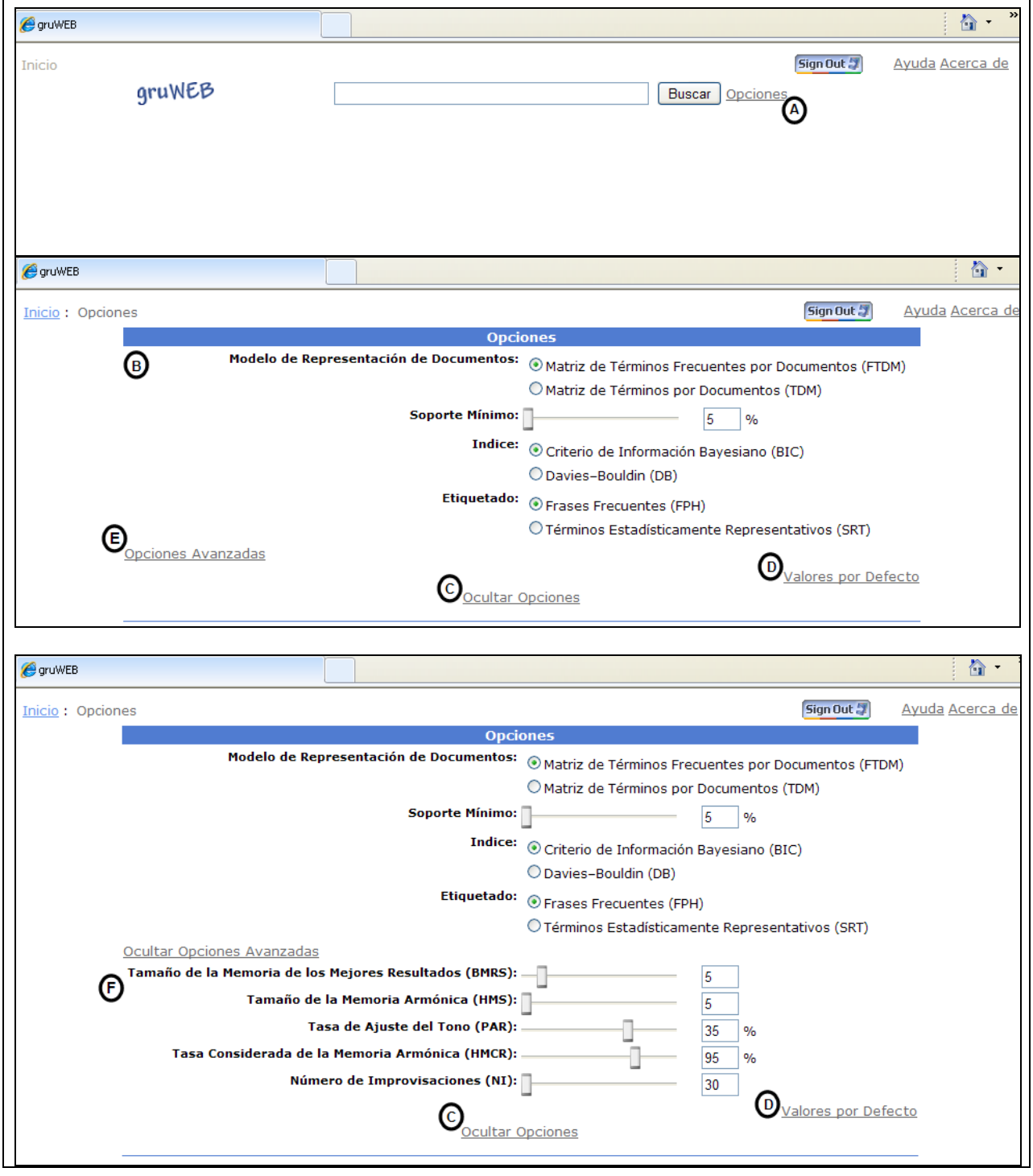
CASO DE USO REAL: INICIAR SESION
Actores: Cliente.
Propósito: El usuario inicia sesión en la aplicación.
Resumen: El usuario inicia sesión en la aplicación, mediante el servicio de Windows Live ID (Passport).
Tipo: Primario.



CURSO NORMAL DE LOS EVENTOS	
Acción del actor	**Respuesta del sistema**
1. El usuario da click en el botón Sign In [A].	2. El sistema lo redirige al servicio se Windows Live.
3. El usuario digita el correo [B], la contraseña [C] y da click en el botón Iniciar sesión [D].	4. El sistema lo redirige nuevamente al meta buscador y habilita la opción de búsqueda.
CURSO ALTERNO	
Acción del actor	**Respuesta del sistema**
2. El usuario no digita los campos obligatorios [A] y [B].	3. El sistema le informa que debe ingresar los campos.

Tabla 9. Caso de Uso Real Iniciar Sesión

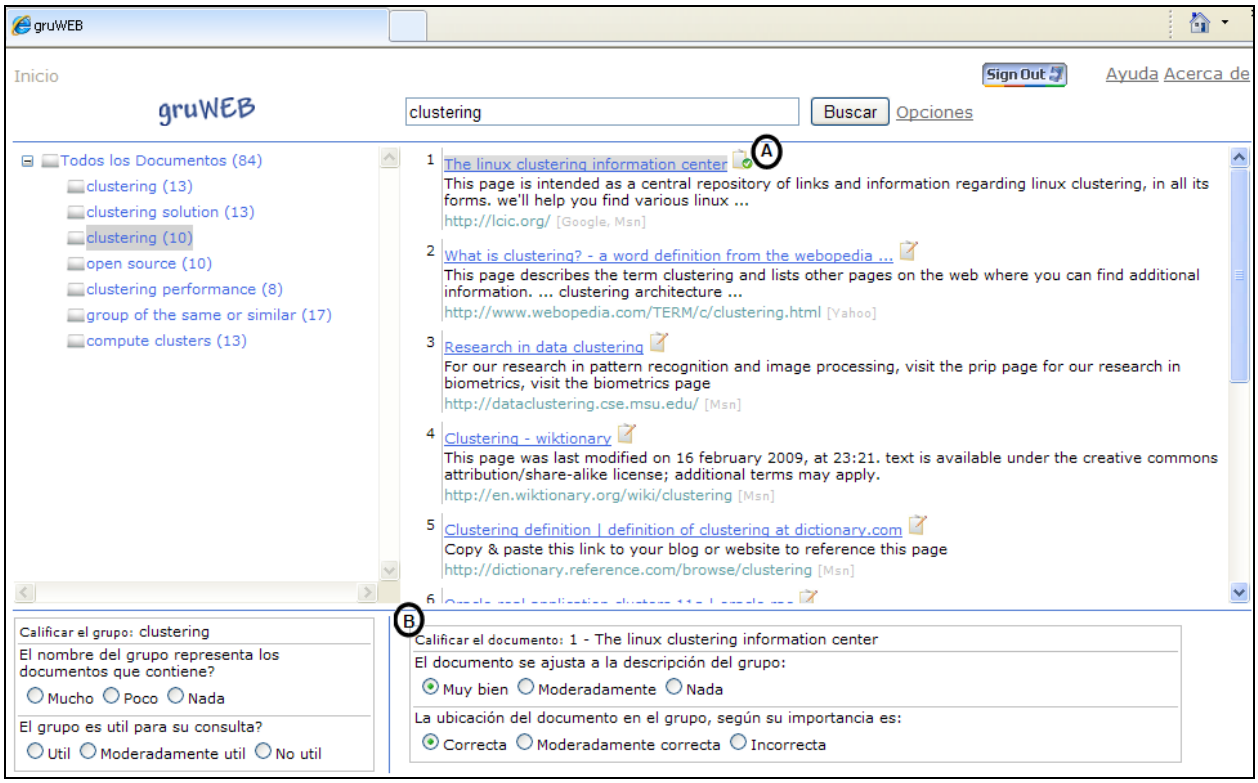
CASO DE USO REAL: ESCOGER PARÄMETROS
Actores: Cliente.
Propósito: Seleccionar los parámetros del algoritmo.
Resumen: El usuario escoge los parámetros deseados para realizar la búsqueda.
Tipo: Secundario.

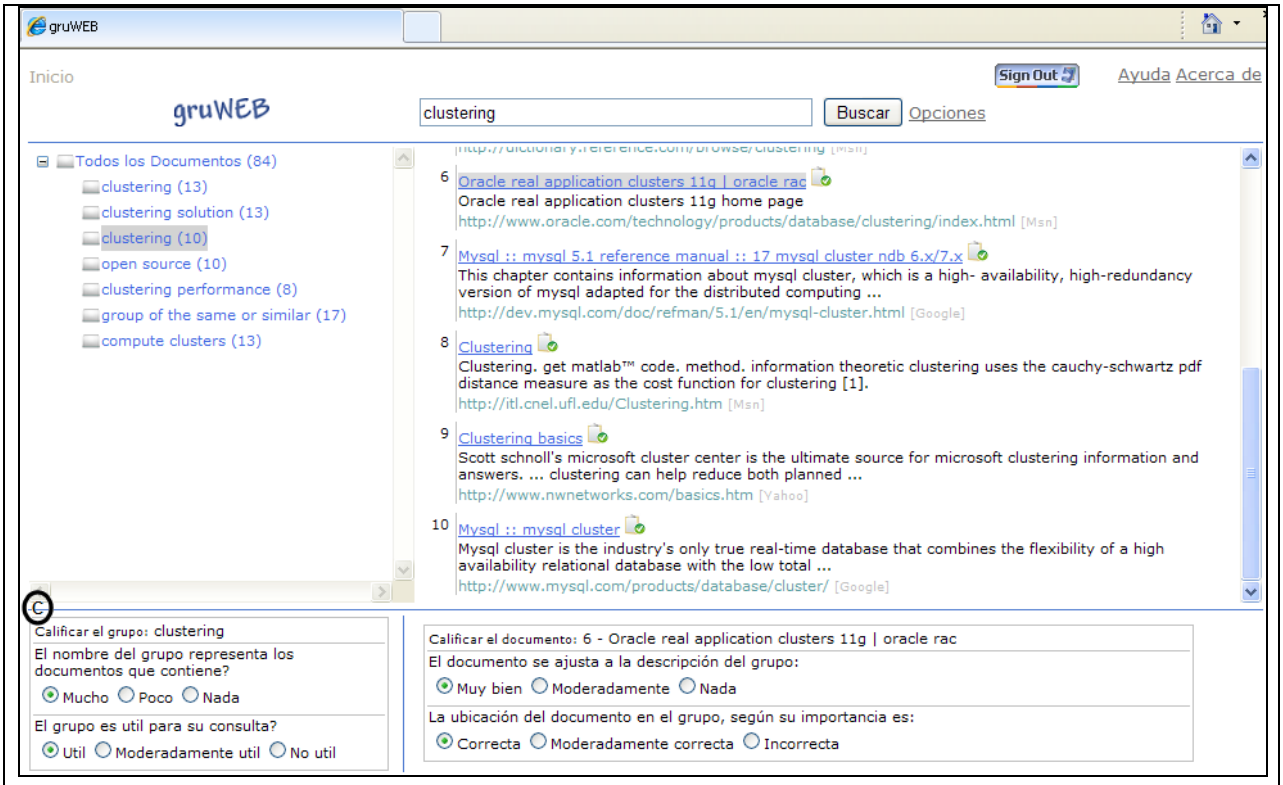


CURSO NORMAL DE LOS EVENTOS	
Acción del actor	**Respuesta del sistema**
1. El usuario da click en el enlace Opciones [A].	2. El sistema muestra al usuario los parámetros que puede modificar.
3. El usuario modifica los parámetros deseados [B], y da click en el enlace Ocultar Opciones [C].	4. El sistema muestra la página principal.
CURSO ALTERNO 1	
Acción del actor	**Respuesta del sistema**
4. El usuario da clic en el enlace Opciones Avanzadas [E].	5. El sistema muestra al usuario los parámetros adicionales que puede modificar.
6. El usuario modifica los parámetros adicionales [F], y da clic en el enlace Ocultar Opciones [C].	7. El sistema muestra la página principal.
CURSO ALTERNO 2	
Acción del actor	**Respuesta del sistema**
8. El usuario da click en el enlace Valores por Defecto.	9. El sistema establece los parámetros a los valores por defecto.
	10. El sistema muestra la página principal.

Tabla 10. Caso de Uso Real Escoger Parámetros

CASO DE USO REAL: CALIFICAR GRUPOS Y DOCUMENTOS
Actores: Cliente.
Propósito: Calificar los grupos y los documentos.
Resumen: El usuario a medida que selecciona un grupo y sus documentos puede realizar la calificación para cada caso.
Tipo: Secundario.





CURSO NORMAL DE LOS EVENTOS	
Acción del actor	**Respuesta del sistema**
1. El usuario da click en el botón de calificación [A], ubicado al lado de título del documento.	2. El sistema le muestra las opciones para calificar el documento [B].
3. El usuario califica el documento que ha seleccionado, según las opciones mostradas en [B].	4. El sistema registra la calificación ingresada en [B] y cambia la imagen del botón de calificación [A].
5. El usuario califica el grupo actual, según las opciones mostradas en [C].	6. El sistema registra la calificación ingresada en [C].

Tabla 11. Caso de Uso Real Calificar Grupos y Documentos

ANEXO H – DIAGRAMA DE CLASES DEL SISTEMA

10. DIAGRAMA DE CLASES DEL SISTEMA

En la Figura 43 se muestra de manera general el diagrama de clases del sistema y en la Tabla 12 se describe la funcionalidad de cada Clase.

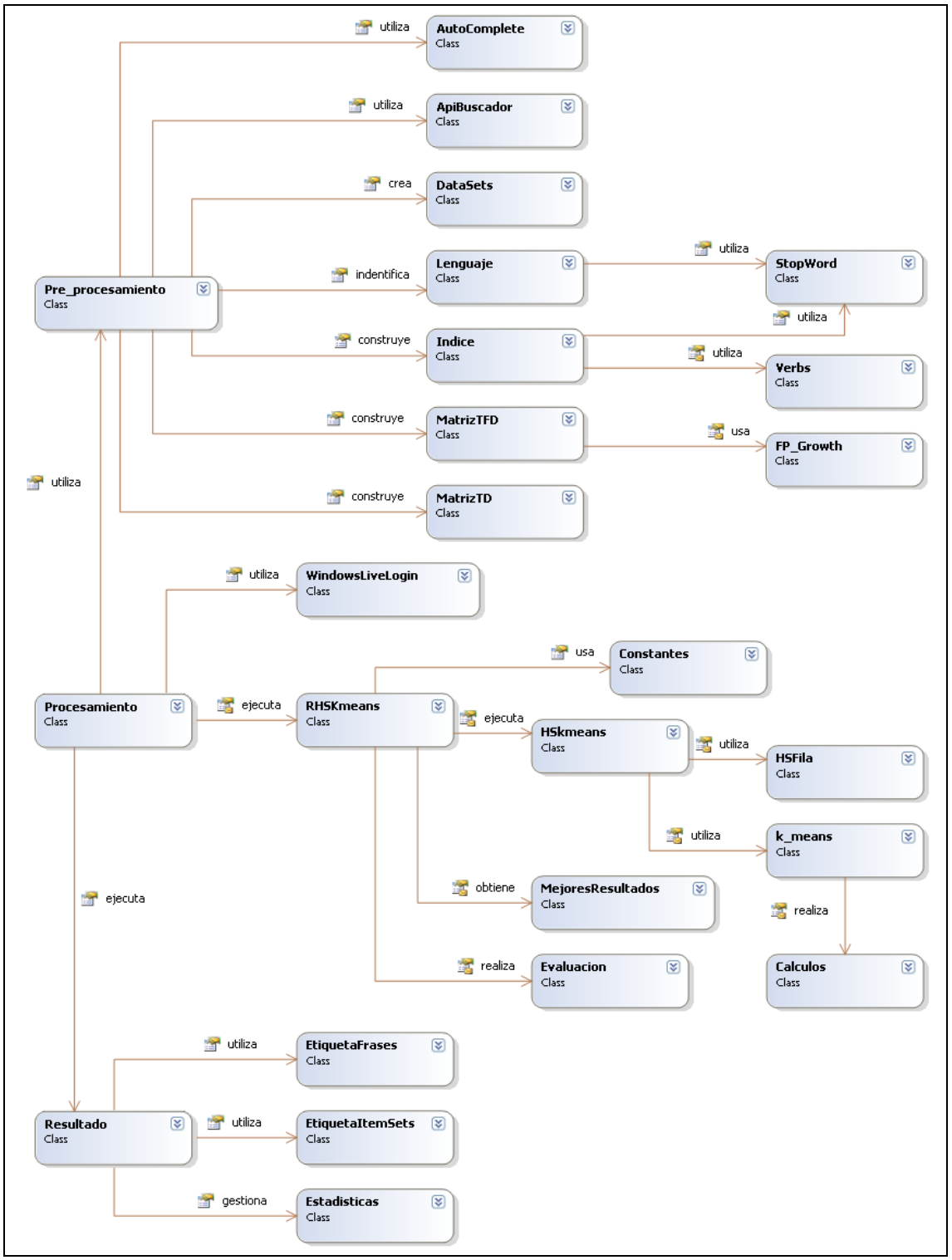


Figura 43. Diagrama General de Clases del Sistema

CLASE	FUNCION
AutoComplete	Funcionalidad provista por Google que autocompleta la consulta en la caja de texto donde se ingresa la consulta, es decir a medida que el usuario escribe la consulta, le aparecen sugerencias de temas asociados.
ApiBuscador	Obtiene y estandariza los documentos provenientes de los buscadores Google, Yahoo! y MSN Live. También provee la funcionalidad provista por Google de traducción, para traducir las etiquetas, los títulos y el resumen del documento.
DataSets	Provee la funcionalidad para crear el Data Set con los documentos.
StopWord	Provee la lista de palabras vacías (de significado) de los diferentes idiomas.
Verbs	Provee la lista de verbos del idioma Inglés.
Lenguaje	Provee la funcionalidad correspondiente a la identificación del lenguaje de los documentos.
Indice	Provee la funcionalidad correspondiente a la indexación de los documentos tales como: análisis léxico, eliminación de palabras vacías y stemming. Igualmente provee la funcionalidad correspondiente a la obtención de las raíces de los documentos para el proceso de etiquetado.
FP-Growth	Obtiene la lista de Itemsets Frecuentes de los documentos.
MatrizTFD	Provee la funcionalidad para crear la Matriz de Términos Frecuentes por Documentos.
MatrizTD	Provee la funcionalidad para crear la Matriz de Términos por Documentos.
Pre-procesamiento	Provee toda la funcionalidad correspondiente al pre-procesamineto de los documentos.
WindowsLiveLogin	Funcionalidad provista por Windows Live Id (servicio Passport), para la autenticación de los usuarios en la aplicación.
Constantes	Mantiene los parámetros para el algoritmo.
Calculos	Provee la funcionalidad para realizar el cálculo de funciones como el Índice BIC, Índice DB, Similitud de Cosenos, entre otras.
k-means	Provee la funcionalidad del algoritmo k-means.
HSFila	Provee la funcionalidad correspondiente a la Creación de Centroides Aleatorios y a la Creación de un Improviso.
HSkmeans	Provee la funcionalidad correspondiente a la creación de una solución del algoritmo IGBHSK.
MejoresResultados	Se encarga de mantener las mejores soluciones del algoritmo IGBHSK.
Evaluacion	Provee la funcionalidad para el cálculo de las medidas de relevancia (Precisión, Exhaustividad y Medida F). Esta clase se utiliza solamente para la evaluación con los DataSets de Reuters y Dmoz.
RHSKmeans	Provee la funcionalidad principal para la ejecución del algoritmo IGBHSK.
EtiquetaFrases	Obtiene las etiquetas de los grupos, para el modelo de Frases Frecuentes.
EtiquetaItemSets	Obtiene las etiquetas de los grupos, para el modelo de Términos Estadísticamente Representativos.
Estadisticas	Provee la funcionalidad para la calificación tanto de los grupos como de

	los documentos.
Resultado	Provee la funcionalidad correspondiente a la asignación de etiquetas, calificación de grupos y documentos, ordenamiento de grupos y documentos.
Procesamiento	Es la clase principal, que procesa la consulta del usuario para obtener los documentos, realizar el pre-procesamiento de los mismos, ejecutar el algoritmo, crear los resultados y procesar la calificación de grupos y documentos.

Tabla 12. Descripción de las Clases

11. CLASES DEL SISTEMA

A continuación se presentan en detalle las clases del sistema (ver Figura 44 a Figura 66).

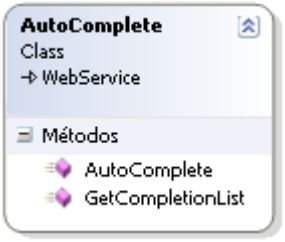


Figura 44. Vista detallada de la Clase AutoComplete

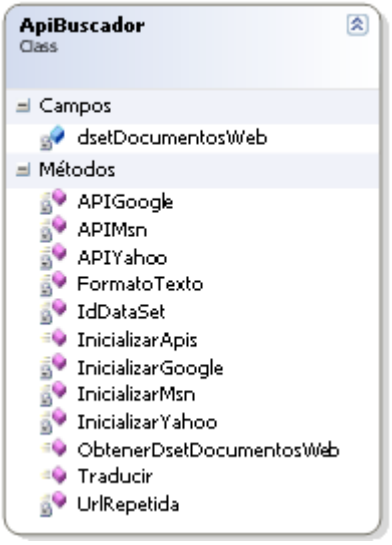


Figura 45. Vista detallada de la Clase ApiBuscador

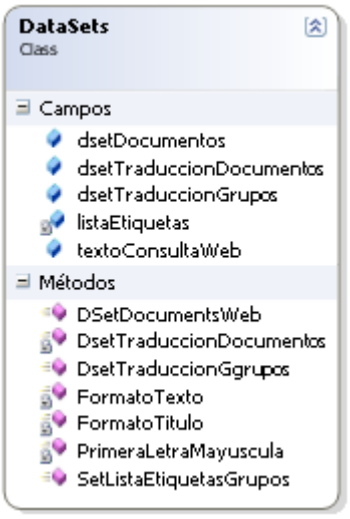


Figura 46. Vista detallada de la Clase DataSets

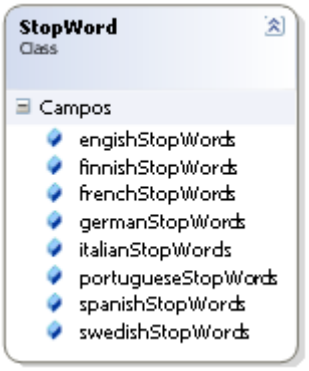


Figura 47. Vista detallada de la Clase StopWord

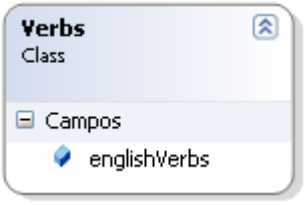


Figura 48. Vista detallada de la Clase Verbs

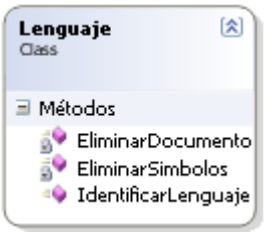


Figura 49. Vista detallada de la Clase Lenguaje

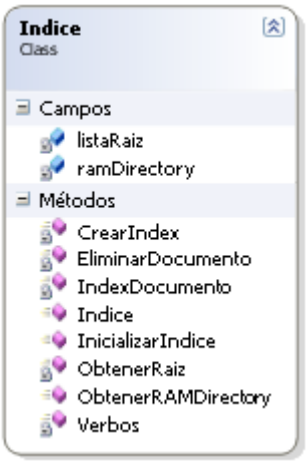


Figura 50. Vista detallada de la Clase Indice

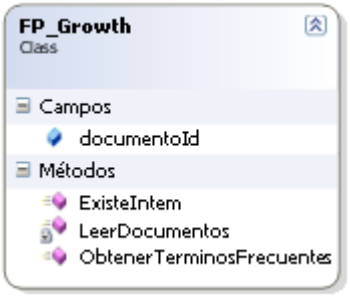


Figura 51. Vista detallada de la Clase FP_Growth

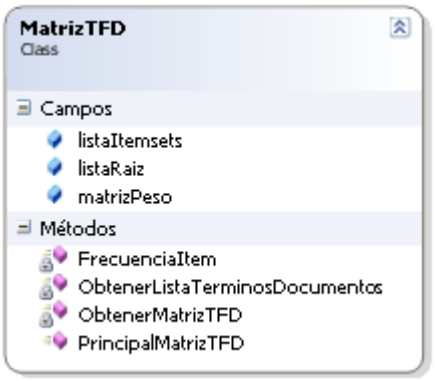


Figura 52. Vista detallada de la Clase MatrizTFD

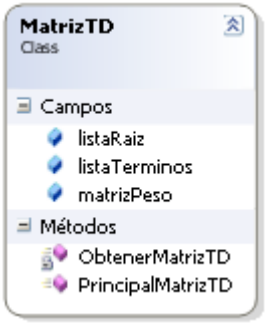


Figura 53. Vista detallada de la Clase MatrizTD

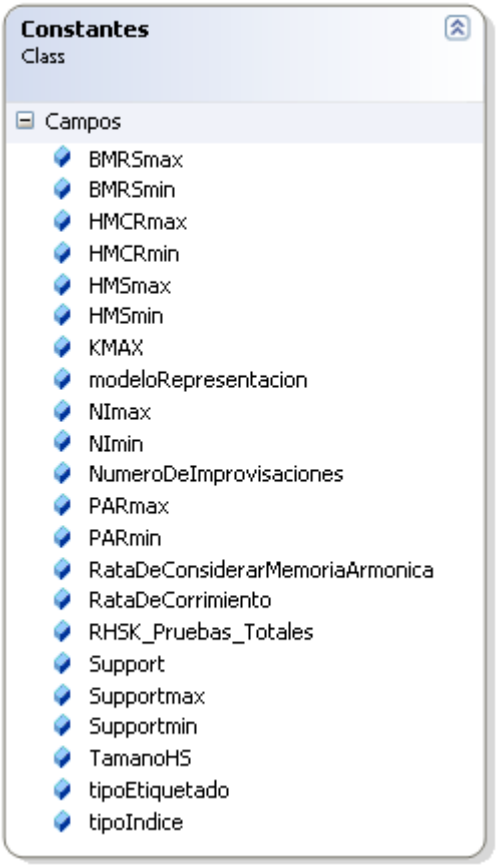


Figura 54. Vista detallada de la Clase Constantes

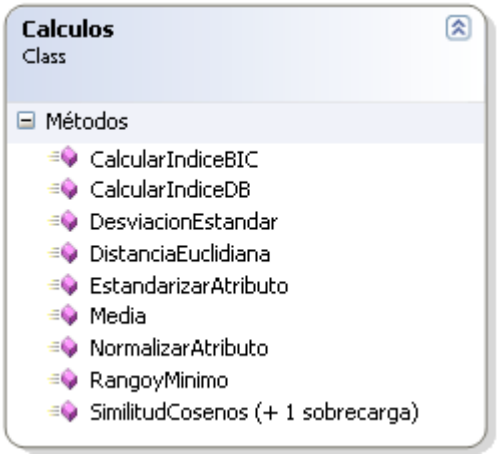


Figura 55. Vista detallada de la Clase Calculos

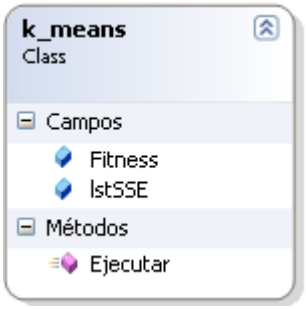


Figura 56. Vista detallada de la Clase k_means

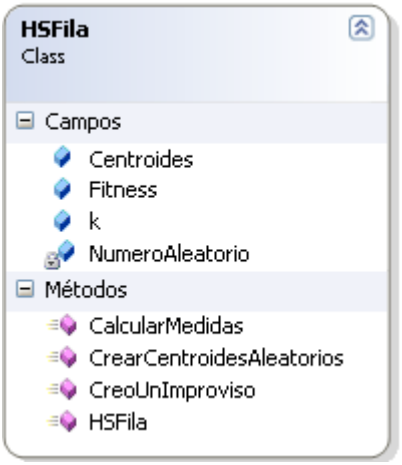


Figura 57. Vista detallada de la Clase HSFila

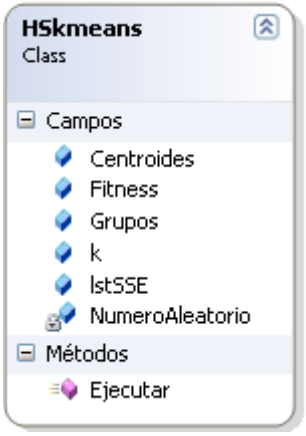


Figura 58. Vista detallada de la Clase HSkmeans

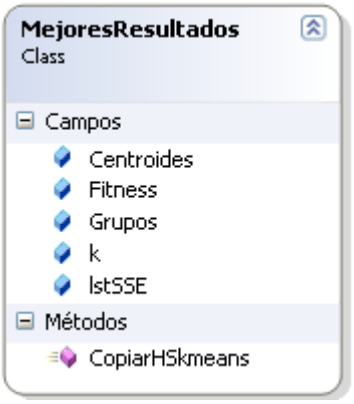


Figura 59. Vista detallada de la Clase MejoresResultados

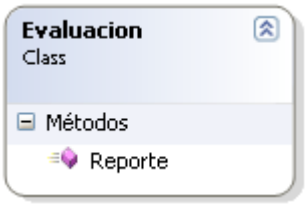


Figura 60. Vista detallada de la Clase Evaluacion

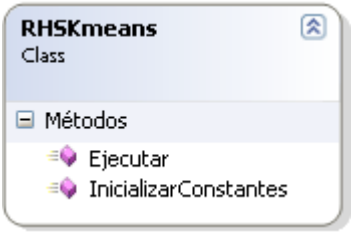


Figura 61. Vista detallada de la Clase RHSKmeans

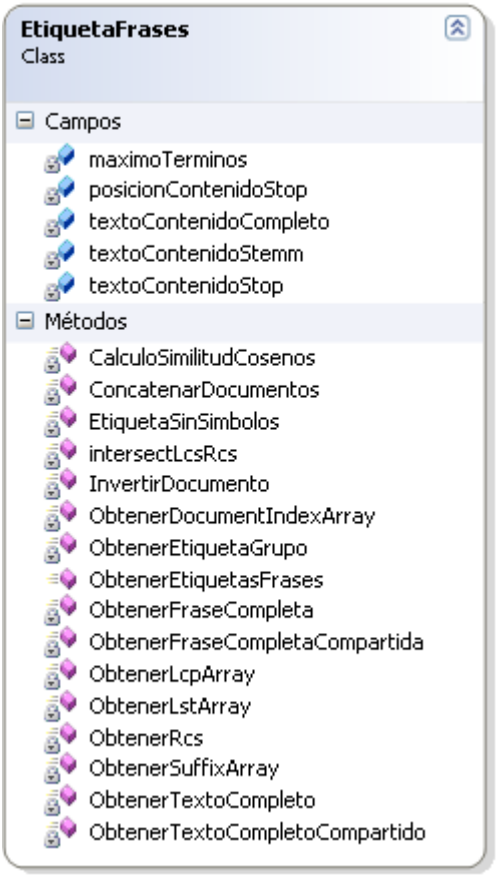


Figura 62. Vista detallada de la Clase EtiquetaFrases

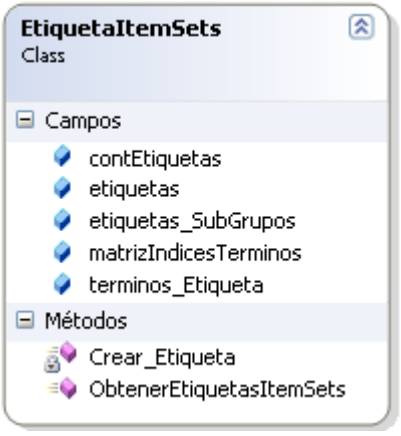


Figura 63. Vista detallada de la Clase EtiquetaItemSets

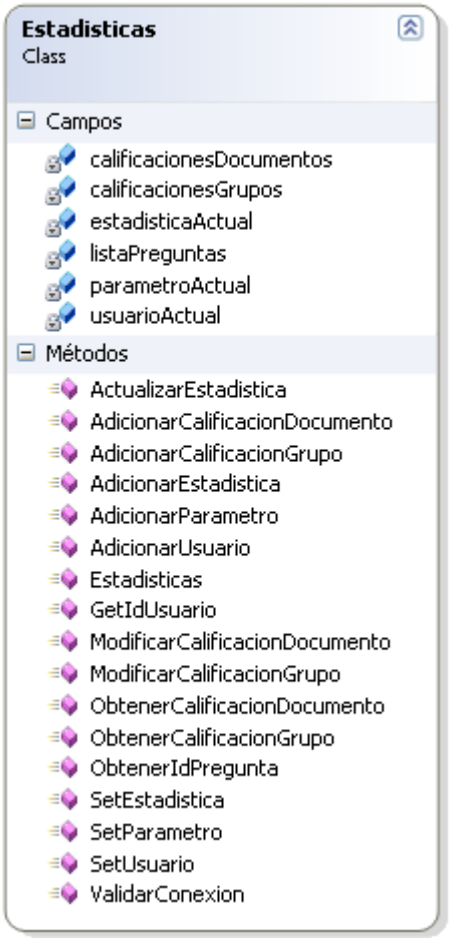


Figura 64. Vista detallada de la Clase Estadisticas

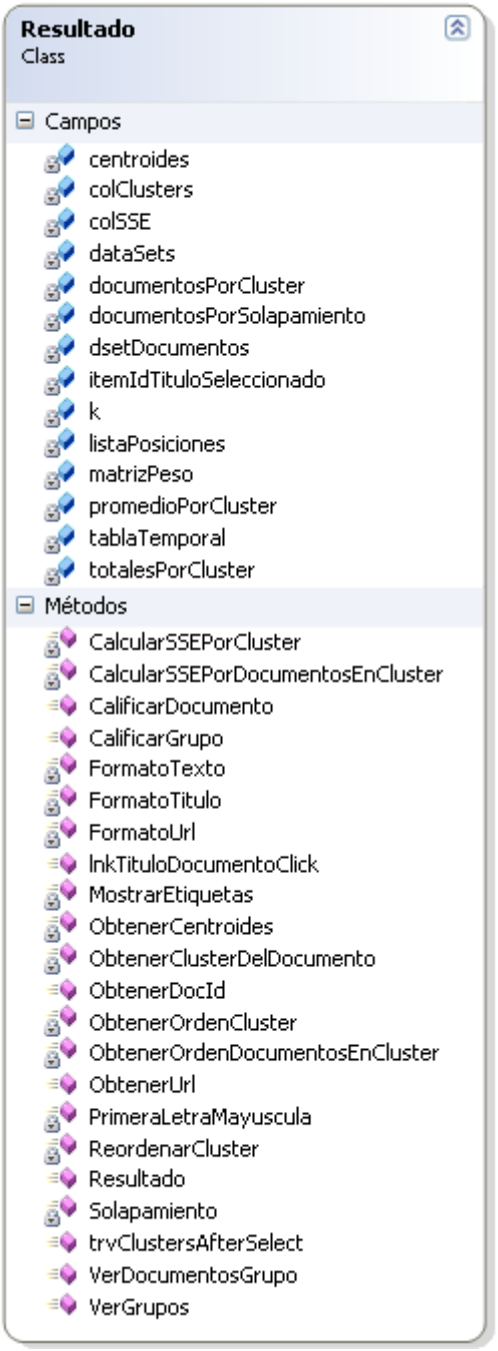


Figura 65. Vista detallada de la Clase Resultado

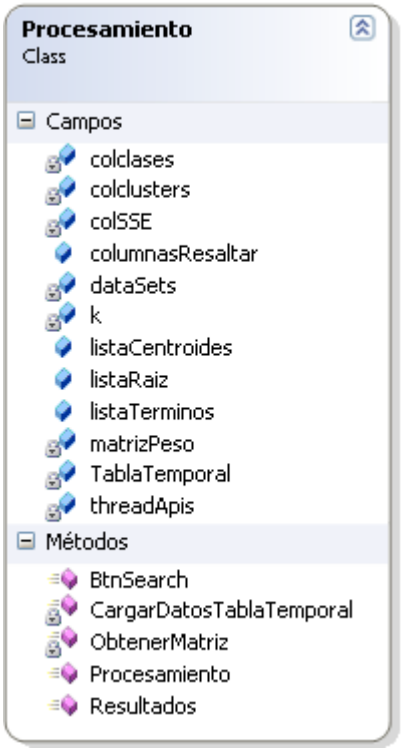


Figura 66. Vista detallada de la Clase Procesamiento

ANEXO I – MANUAL DE USUARIO

12. MANUAL DE USUARIO

GruWeb es un meta buscador que permite realizar búsquedas en la web y observar los resultados en grupos de documentos, cada grupo identificado con un nombre. A continuación se explica en detalle cómo se puede utilizar el meta buscador.

Una vez que se ha accedido a la página principal, se observa la opción de búsqueda deshabilitada debido a que el usuario debe primero iniciar sesión utilizando una cuenta de Windows Live ID, como se observa en la Figura 67.

Figura 67. Página principal del meta buscador GruWeb

En la página principal podemos observar en la parte superior derecha dos enlaces sobre "Ayuda" y "A cerca de", los cuales nos ofrecen información sobre el uso del meta buscador y el desarrollo de la herramienta respectivamente, como se observa en las Figura 68 a Figura 70.

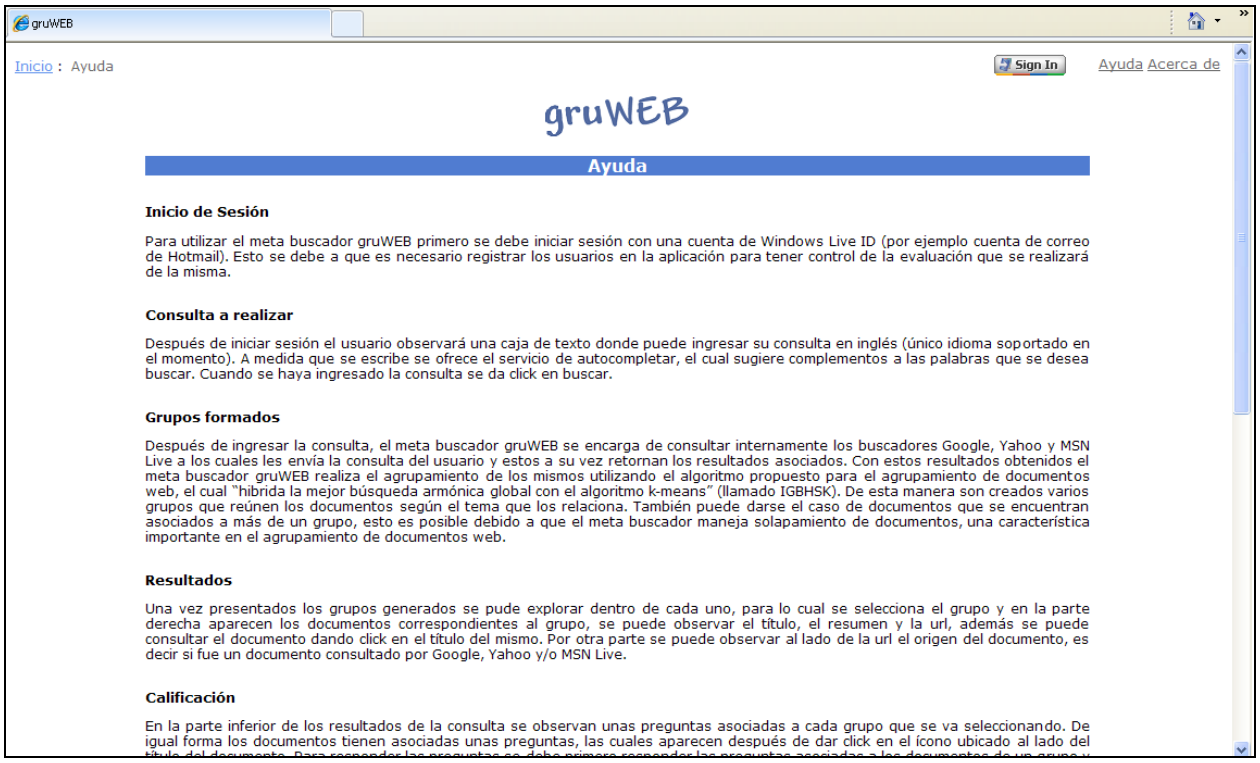


Figura 68. Ayuda del meta buscador GruWeb (primera parte)

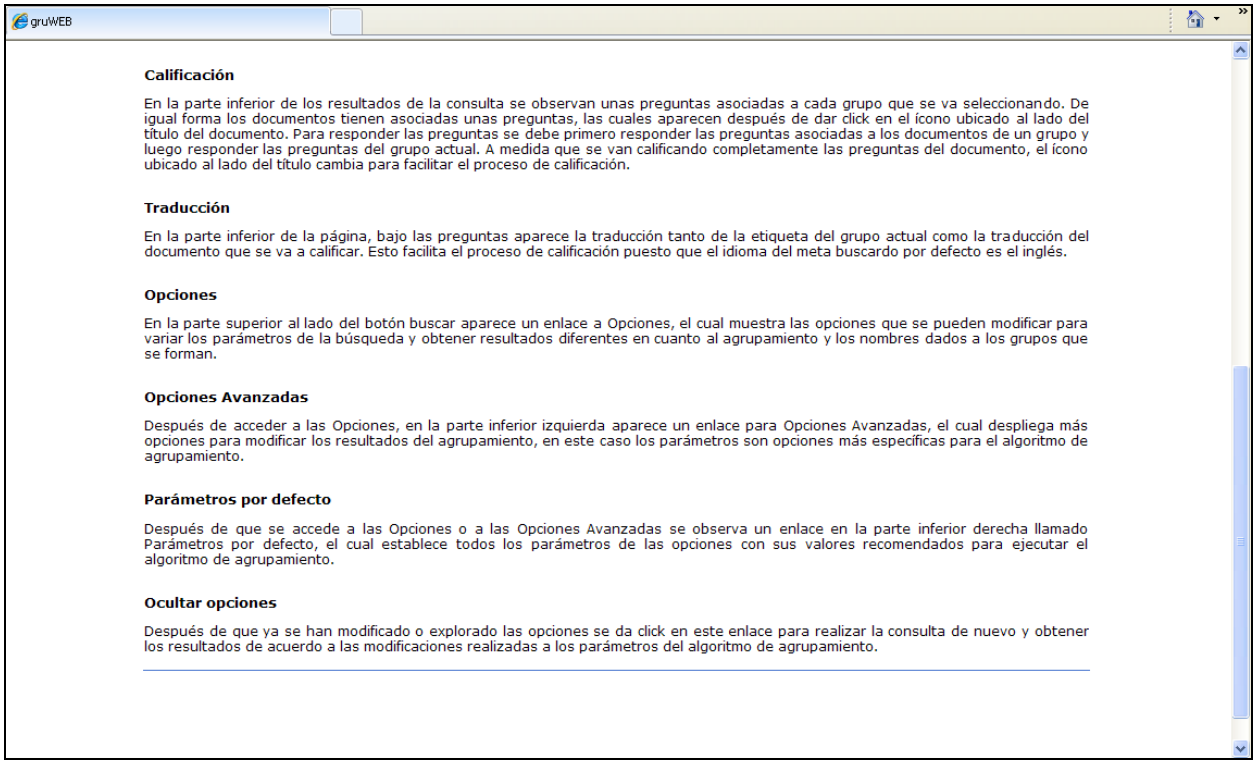


Figura 69. Ayuda del meta buscador GruWeb (segunda parte)

Figura 70. A cerca del meta buscador GruWeb

Como se mencionó anteriormente para comenzar a buscar utilizando GruWeb, primero se debe iniciar sesión con Windows Live ID, como se observa en las Figura 71 y Figura 72. Esto se debe a que es necesario registrar los usuarios en la aplicación para tener control de la evaluación que se realizará de la misma.

Figura 71. Inicio de Sesión utilizando Windows Live ID

Figura 72. Login y Password de Windows Live ID

Una vez sea ha iniciado la sesión en Windows Live ID, el meta buscador GruWeb está listo para ser utilizado en las búsqueda a la web, como se observa en la Figura 73.

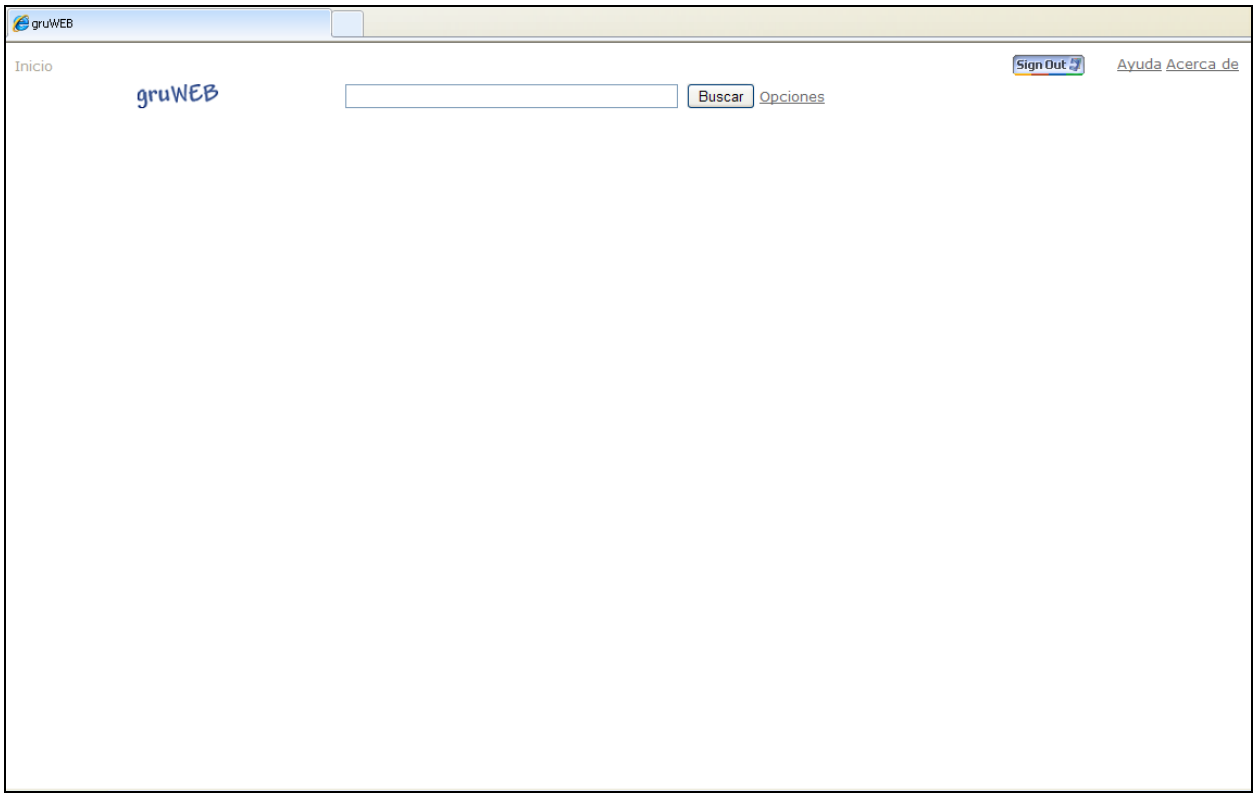


Figura 73. Búsqueda habilitada en GruWeb con sesión abierta en Windows Live ID

Antes de ingresar una consulta el usuario puede acceder al enlace de opciones ubicado al lado del botón buscar, este enlace le permite modificar las opciones (básicas y avanzadas) de búsqueda que internamente utiliza el meta buscador para obtener diferentes resultados en el agrupamiento de los documentos y en los nombres dados a los grupos que se forman, como se observa en la Figura 74 y Figura 75. Así mismo en la parte inferior se observa un enlace llamada “Parámetros por Defecto”, el cual establece todos los parámetros de las opciones con sus valores recomendados para ejecutar la aplicación.

Después de que se han modificado o explorado las opciones de búsqueda se da click en el enlace “Ocultar Opciones”, que se encuentra en la parte inferior. Este enlace permite realizar la consulta de nuevo y obtener los resultados de acuerdo a las modificaciones realizadas a los parámetros del meta buscador.

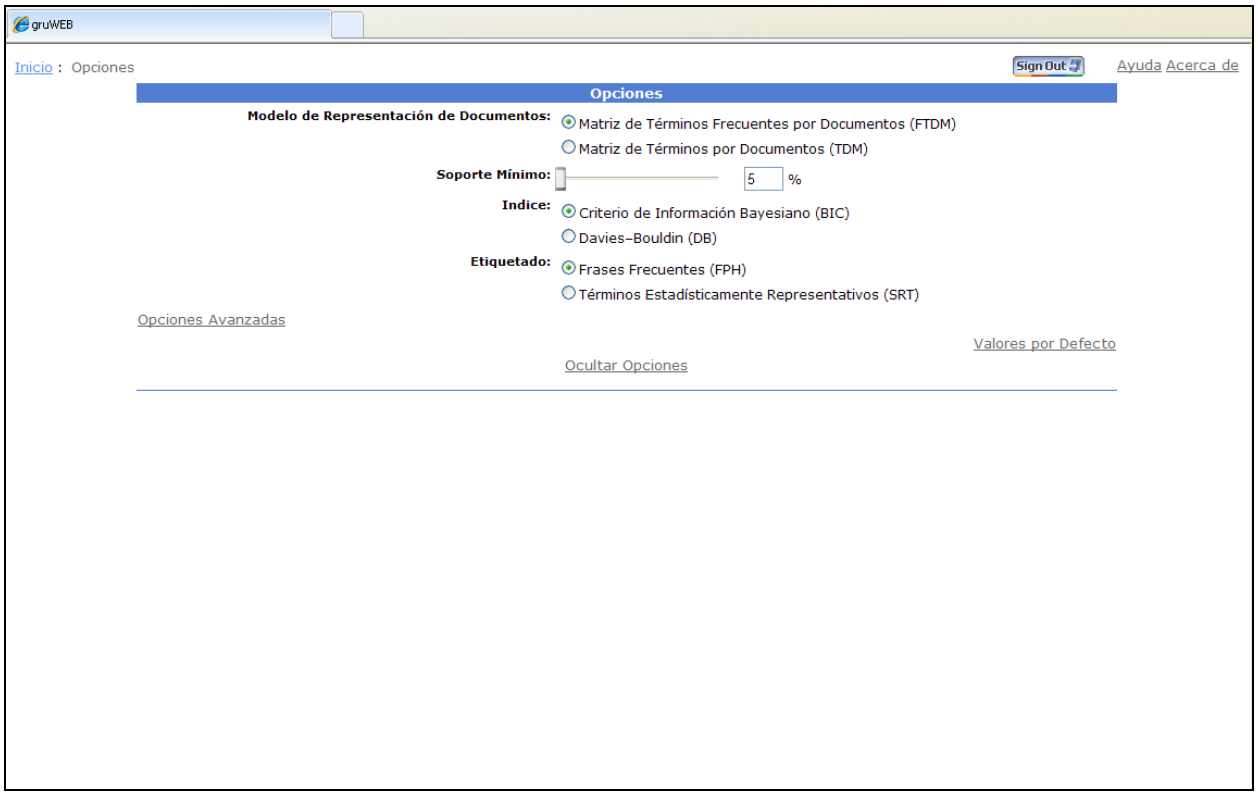


Figura 74. Opciones básicas que se pueden escoger en el meta buscador GruWeb

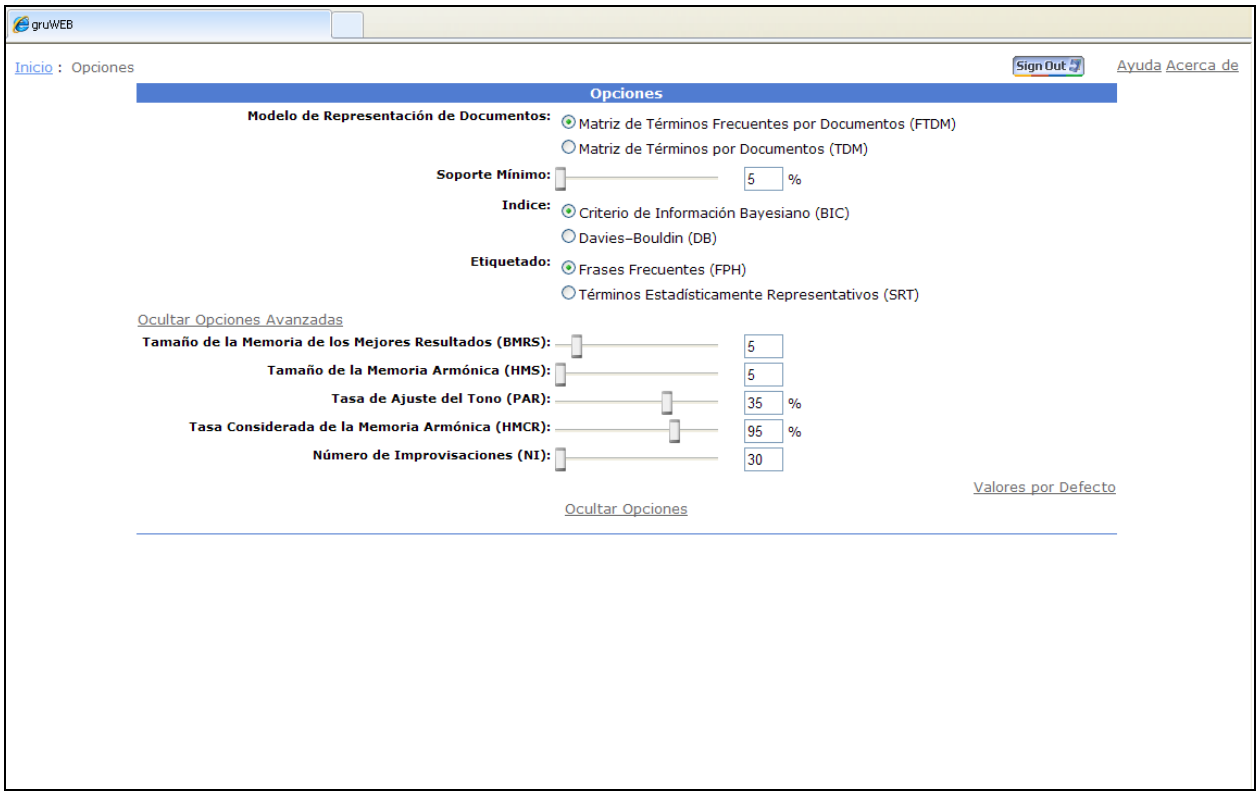


Figura 75. Opciones avanzadas que se pueden escoger en el meta buscador GruWeb

Luego de haber modificado las opciones del meta buscador se ingresa la consulta a realizar, como se observa en la Figura 76.

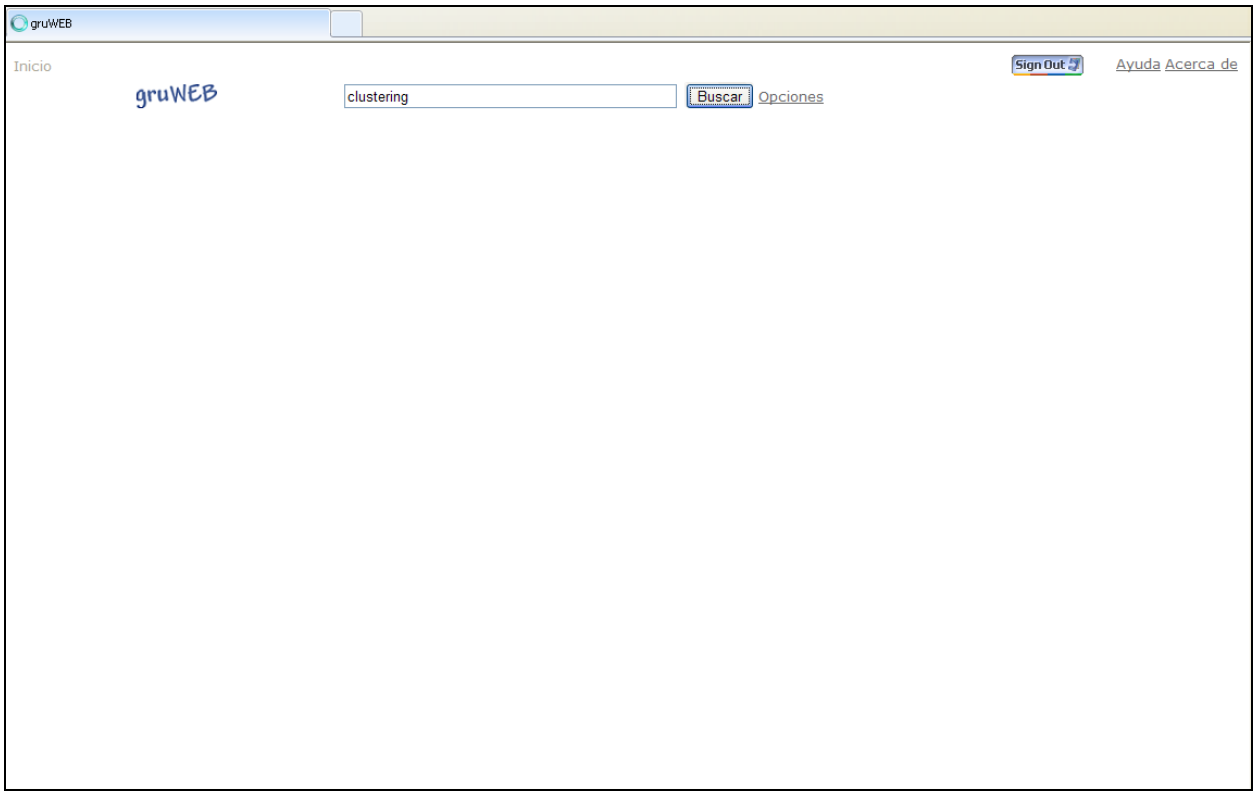


Figura 76. Consulta a realizar en el meta buscador GruWeb

Después de realizar la búsqueda, el meta buscador GruWeb presenta los resultados obtenidos como se observa en la Figura 77. En la parte izquierda se observan los grupos de documentos que formó el meta buscador, y en la parte derecha se presentan todos los documentos que consultó relacionados con la consulta ingresada. Si se selecciona algún grupo en especial en la parte derecha aparecerán solo los documentos asociados al grupo seleccionado.

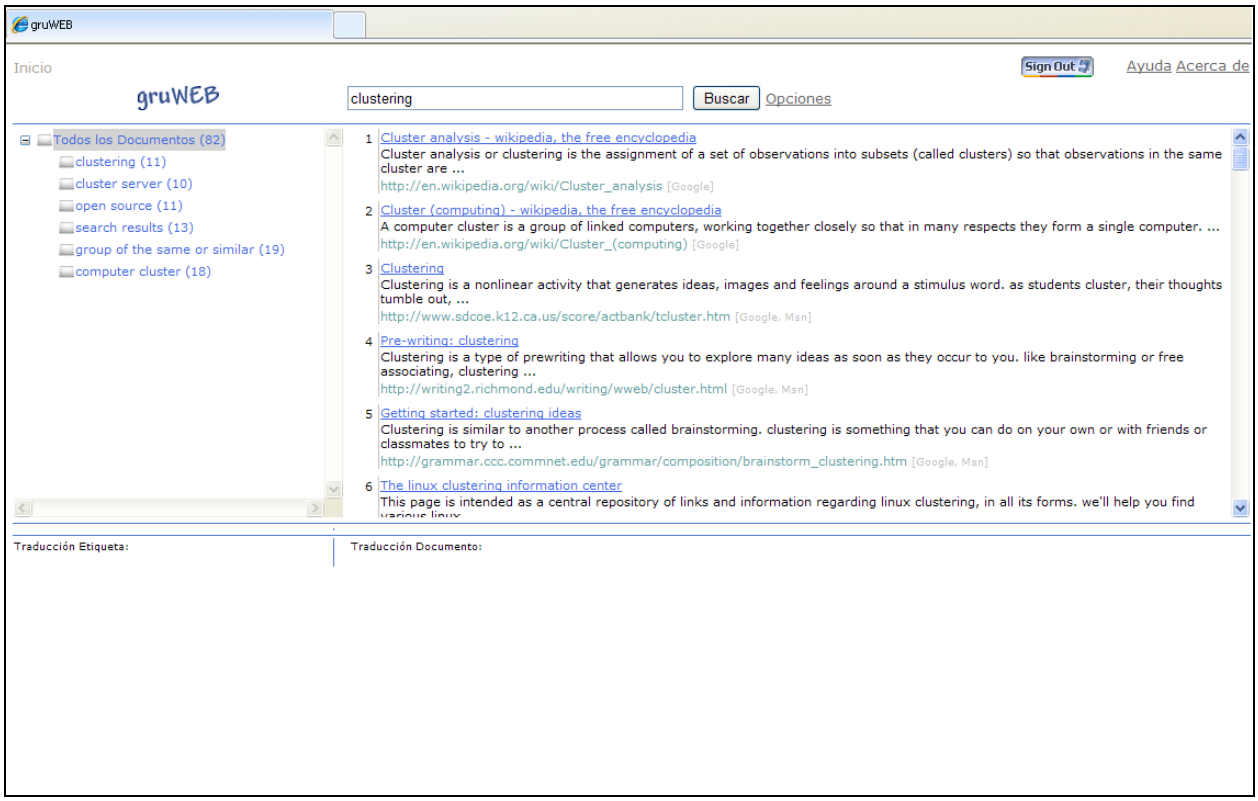


Figura 77. Resultados de una consulta en el meta buscador GruWeb

Cuando se selecciona un grupo además de aparecer en el lado derecho sus documentos asociados también aparece en la parte inferior izquierda una sección que permite calificar el grupo que se ha seleccionado, para ello se deben responder las dos preguntas asociadas al grupo. Se recomienda que estas preguntas se respondan luego de haber observado los documentos en el grupo, como se observa en la Figura 78. Adicionalmente, se presenta debajo de la sección de calificación la traducción del nombre de la etiqueta, esto facilitará la calificación por parte de los usuarios. Por otra parte es posible que un documento aparezca asociado a más de un grupo, esto se debe a que maneja solapamiento de documentos, aspecto importante para el usuario en el clustering de documentos.

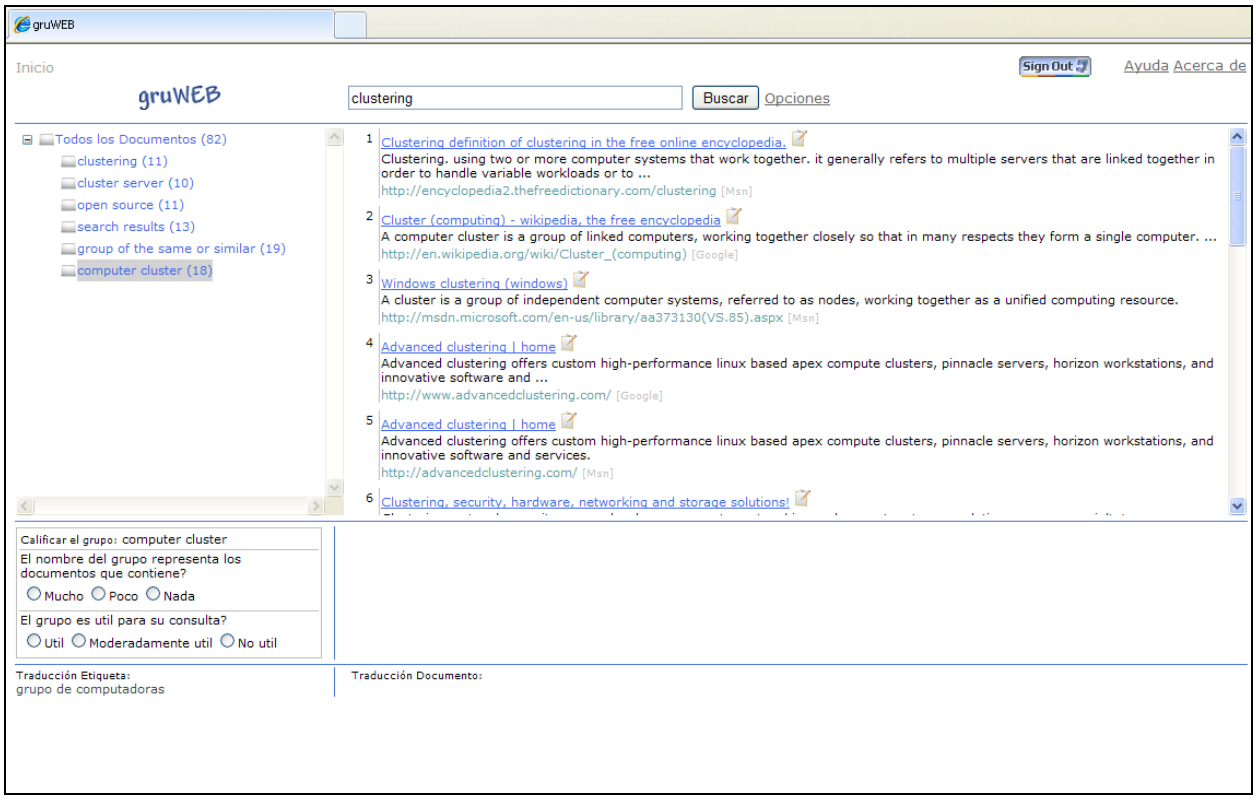


Figura 78. Calificación del Grupos en el meta buscador GruWeb

Cuando se observan los documentos de cada grupo, se aprecia al lado del título de cada documento un ícono que permite realizar la calificación del documento, para ello se da click sobre el ícono y a continuación aparecen asociadas a los documentos dos preguntas en la parte inferior derecha, como se observa en la Figura 79. Adicionalmente, se observa de bajo de la calificación la traducción del título del documento y la traducción del snippet (o resumen) del documento.

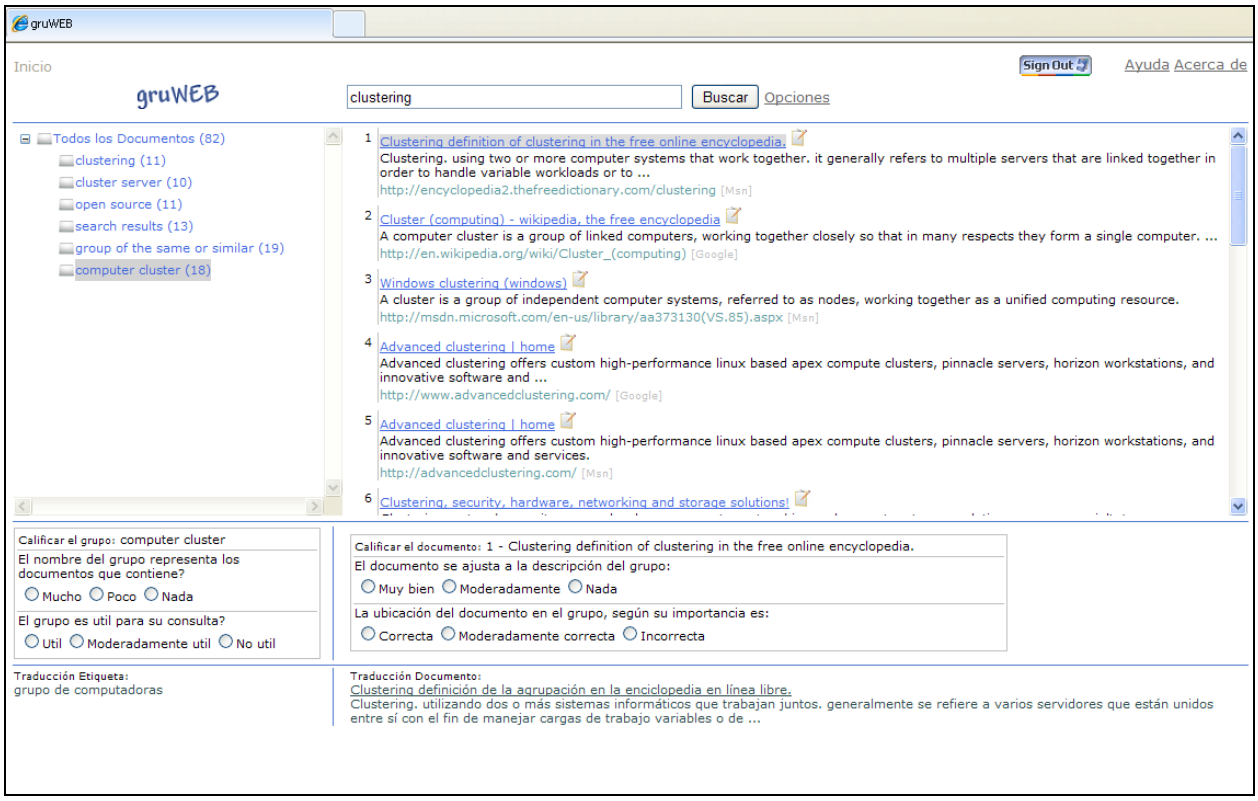


Figura 79. Calificación de los documentos en el meta buscador GruWeb

Cuando se califican las dos preguntas de un documento, el ícono que permite calificar el documento cambia, como se observa en la Figura 80.

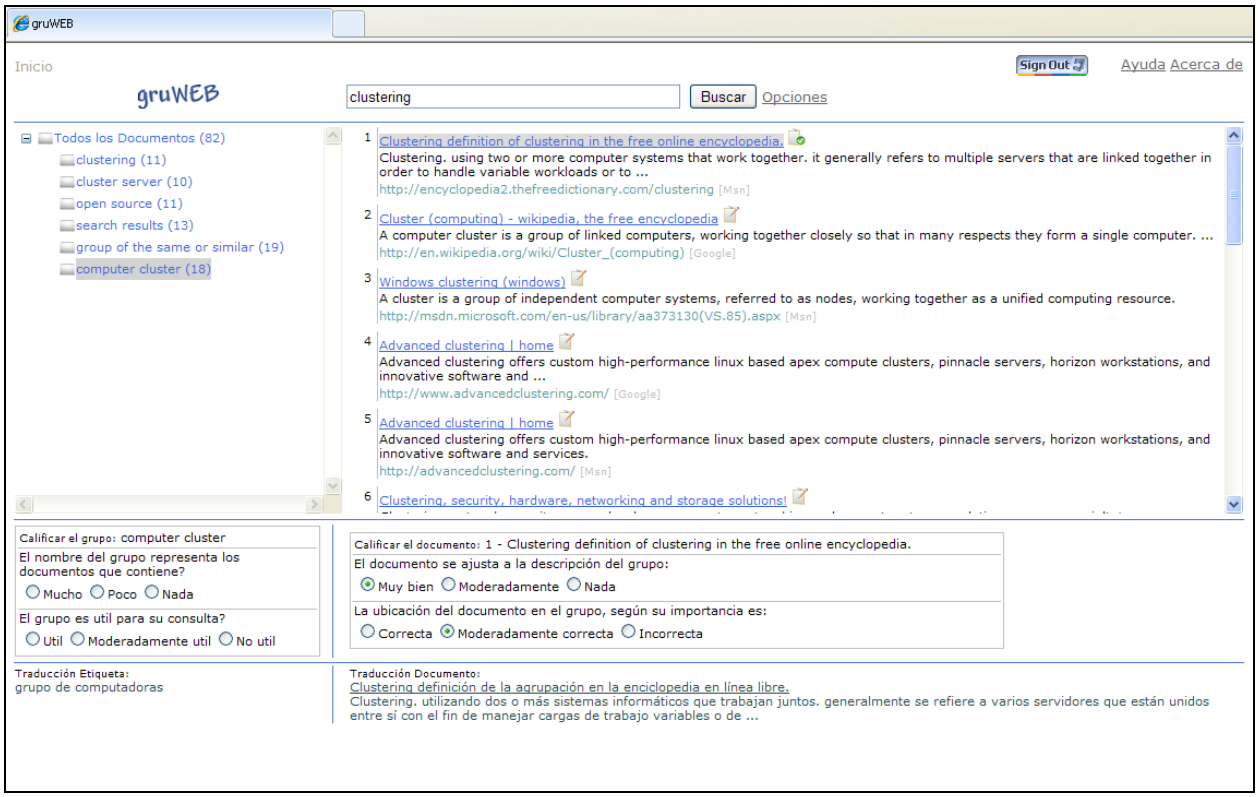


Figura 80. Calificación completa de un documento en el meta buscador GruWeb

Si se desea ver el contenido de algún documento se da click sobre el título del documento y este se abre en la misma página, como se observa en la Figura 81.

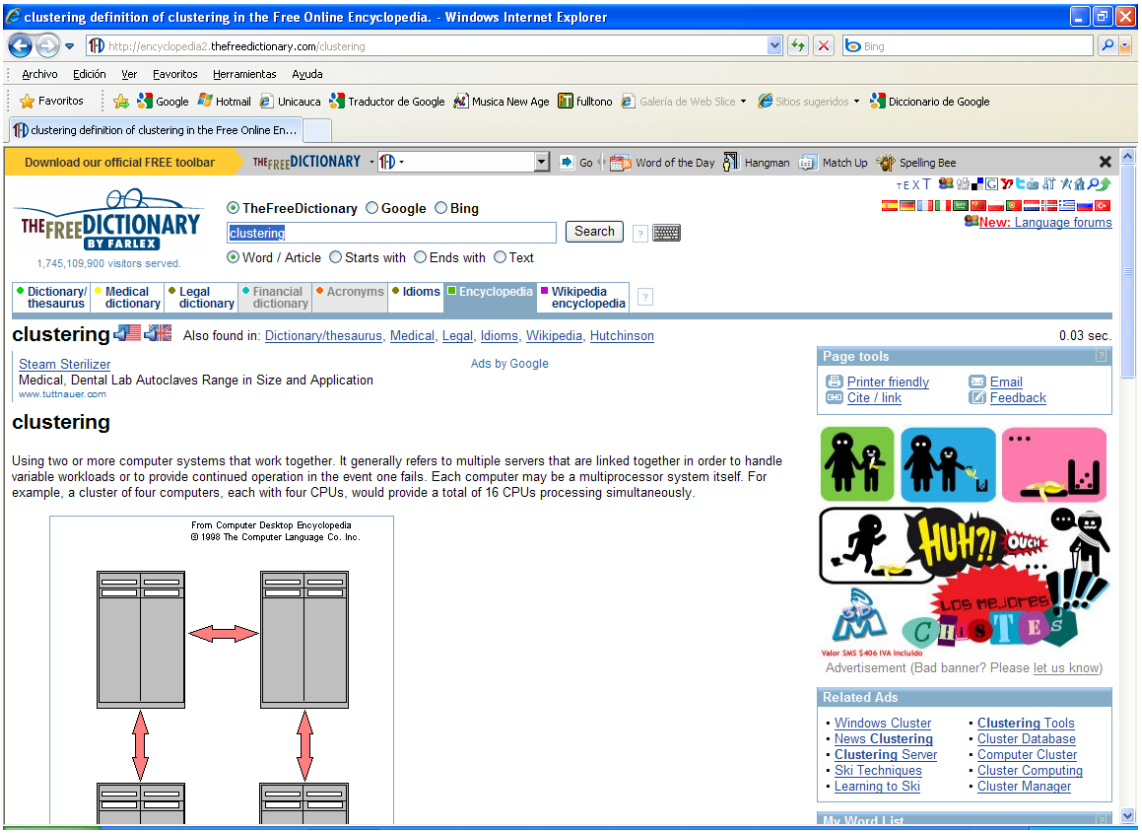


Figura 81. Vista de un documento accedido desde el meta buscador GruWeb

ANEXO J – ARTICULO

13. ARTICULO PRESENTADO A IEEE CEC 2010

Se escribió un artículo titulado: "*Web document clustering based on Global-Best Harmony Search, K-means, Frequent Term Sets and Bayesian Information Criterion*", el cual se presentó al evento internacional denominado 2010 IEEE Congress on Evolutionary Computation () a realizarse en Barcelona, España del 18 al 23 de julio de 2010. Sitio web: http://wcci2010.org/. Actualmente el artículo se encuentra en proceso de evaluación.

A continuación se presenta el contenido del artículo tal como se presentó al evento IEEE CEC 2010.

Web document clustering based on Global-Best Harmony Search, K-means, Frequent Term Sets and Bayesian Information Criterion

Carlos Cobos, Jennifer Andrade, William Constain, Martha Mendoza, Elizabeth León

***Abstract* — This paper introduces a new description-centric algorithm for web document clustering based on the hybridization of the Global-Best Harmony Search with the K-means algorithm, Frequent Term Sets and Bayesian Information Criterion. The new algorithm defines the number of clusters automatically. The Global-Best Harmony Search provides a global strategy for a search in the solution space, based on the Harmony Search and the concept of swarm intelligence. The K-means algorithm is used to find the optimum value in a local search space. Bayesian Information Criterion is used as a fitness function, while FP-Growth is used to reduce the high dimensionality in the vocabulary. This resulting algorithm, called IGBHSK, was tested with data sets based on Reuters-21578 and DMOZ, obtaining promising results (better precision results than a Singular Value Decomposition algorithm). Also, it was also then evaluated by a group of users.**

I. Introduction

In recent years, web document clustering has become a very interesting research field. This is an alternative presentation of results based on what is known as the cluster hypothesis [6], according to which the clustering of documents may be beneficial to users of an information retrieval system, since it is likely that the results relevant to the user are close to each other in the document space, and therefore tend to fall into a relatively reduced number of clusters [7] allowing reductions in the search time.

To obtain good results in web document clustering the algorithms must meet the following specific requirements [8, 9]: Automatically define the number of clusters that are going to be created; generate relevant clusters for the user and assign these documents to appropriate clusters; define labels or names for the clusters that are easily understood for system users; handle overlapping clusters (this means that documents can belong to multiple clusters); reduce the high dimension that is presented in the management of document collections; handle the processing time, which means for example that the algorithm must be able to work with snippets and not only with the full text of the document; and handle the noise that is very common in the collection of documents.

Another important aspect when studying or proposing an algorithm to perform web document clustering is the document representation model. The most widely used models are [10]: ***Vector space model*** [6, 11], in which the documents are designed as bags of words, the document collection is represented by a matrix of D-terms by N-documents, each document is represented by a vector of normalized frequency term (tf_i) by the document inverse frequency for that term, in what is known as TF-IDF value, and the cosine distance is used for measuring the degree of similarity between two documents or between a document and the user's query. A process of stop word removal and stemming [6] should be done before re-presenting the document.

Several algorithms for web document clustering already exist, but results show there is still room for much to be done. These algorithms, by example, report precision and recall values between only 0.6 and 0.8, when the goal is 1.0 and their cluster labels are confused. This is the main motivation of the present work, in which a new algorithm that obtains better results for web document clustering is proposed.

The remainder of the paper is organized as follows. Section 2 presents some related work, the Global-Best Harmony Search algorithm and the K-means clustering algorithm. The proposed new algorithm is described in detail in Section 3. Section 4 shows the experimental results. Finally, some concluding remarks and suggestions for future work are presented.

II. Related Work

In general, clustering algorithms can be classified into [12], [13]: hierarchical, partitional, density-based, grid-based, and model-based algorithms, among others. The algorithms most commonly used for web document clustering have been the hierarchical and the partitional ones [11]. The hierarchical algorithms generate a dendogram or a tree of groups. This tree starts from a similarity measure, among which are: single link, complete link and average link. In relation to web document clustering, the hierarchical algorithm that brings the best results in accuracy is called UPGMA (Unweighted Pair-Group Method using Arithmetic averages) [12], [14].

In partitional clustering, the algorithms perform an initial division of the data in the clusters and then move the objects from one cluster to another based on the optimization of a

Manuscript received January 30, 2010. This work was supported by a Research Grant from the University of Cauca under Project VRI-2560 and the National University of Colombia (Bogotá).
Carlos Cobos is with University of Cauca (phone: 57-2-8209800x2119; fax: 57-2-8209800x2102; e-mal: ccobos@unicauca.edu.co).
Jennifer Andrade, William Constain and Martha Mendoza are with University of Cauca (e-mal: {jandrade, wconstain, mendoza}@unicauca.edu.co).
Elizabeth León is with National University of Colombia (e-mail: eleonguz@unal.edu.co).

predefined criterion or objective function [13]. The most representative algorithms using this technique are: K-means, K-medois, and Expectation Maximization. In 2000, a Bisecting K-means [11, 15] algorithm was devised. This algorithm combines the strengths of the hierarchical and partitional methods reporting better results concerning the accuracy and the efficiency of the UPGMA and the K-means algorithms.

In 1998 the first algorithm to take the approach based on frequent phrases shared by documents in the collection was put forward, called Suffix Tree Clustering (STC) [9]. Later in 2001, the SHOC (Semantic, Hierarchical, Online Clustering) algorithm was introduced [16]. SHOC improves STC and is based on LSI and frequent phrases. Next in 2003, the Lingo algorithm [17-22] was devised. This algorithm is used by the Carrot2 web searcher and it is based on complete phrases and LSI with Singular Value Decomposition (SVD). Lingo is an improvement of SHOC and STC, and unlike most of the algorithms, tries first to discover descriptive names for the clusters and only then organizes the documents into appropriate clusters. Also in 2007, the Dynamic SVD clustering (DSC) [7] algorithm was made available. This algorithm uses SVD and minimum spanning tree (MST). This algorithm has better performance than Lingo. Finally in 2008, the CFWS (Clustering based on Frequent Word Sequences) and the CFWMS (Clustering based on Frequent Word Meaning Sequences) [15] algorithms were proposed. These algorithms represent text documents as frequent word sequences and frequent concept sequences (based on WordNet), respectively.

In relation to a frequent word sets model for web document clustering, in 2002, FTC (Frequent Term-Based Text Clustering) and HFTC (Hierarchical Frequent Term-Based Text Clustering) algorithms became available [23]. These algorithms use combinations of frequent words (association rules approach) shared in the documents to measure their proximity in the text clustering process. Then in 2003, FIHC (Frequent Itemset-based Hierarchical Clustering) was introduced [24] which measures the cohesion of a cluster using frequent word sets, so that the documents in the same cluster share more frequent word sets than those in other groups. These algorithms provide accuracy similar to that reported for Bisection K-means, with the advantage that they assign descriptive labels to associate clusters.

Finally, in partitional clustering from an evolutionary approach, in 2007 three hybridization methods between the Harmony Search (HS) [25] and the K-means algorithms [26] were compared. These were: Sequential hybridization method, interleaved hybridization method and the hybridization of K-means as a step of HS. As a general result, the last method was the best choice of the three. Later in 2008 [27, 28] [25], based on the Markov Chains theory the researchers demonstrated that the last algorithm converges to the global optimum. This proposal is a data-centric algorithm [8], it does not define the number of clusters automatically and it does not show appropriate cluster labels. Next in 2009, a Self-Organized Genetic [29] algorithm was devised for text clustering based on the WordNet ontology. In this algorithm, a modified LSI model was also presented, which appropriately gathers the associated semantic similarities. This algorithm outperforms the standard genetic algorithm [30] and the K-means algorithm for web document clustering in similar environments.

A. Global-Best Harmony Search (GBHS) Algorithm

Harmony Search (HS) is a meta-heuristic algorithm mimicking the improvisation process of musicians (where music players improvise the pitches of their instruments to obtain better harmony) [25, 28, 31]. HS has been successfully applied to many optimization problems: travelling salesman problem [4], power economic dispatch [29], synchronization of discrete-time chaotic systems [32], and for web document clustering [28], among others. Global-Best Harmony Search [33] is a new variant of HS. GBHS is inspired by the concept of swarm intelligence as proposed in Particle Swarm Optimization (PSO) [34]. The steps in the GBHS algorithm can be summarized as shown in **Fig 1**.

```
01   Initialize the Problem and Algorithm Parameters
02   Initialize the Harmony Memory (HM)
     Repeat
03       Improvise a New Harmony (best from memory)
04       Update the Harmony Memory
05   Until (Stop Criterion)
06   Return best solution
```

Fig 1. The GBHS algorithm

In step 03, a New Harmony vector is generated based on three rules: memory consideration (best harmony in the HM), pitch adjustment and random selection. In step 04, the New Harmony vector replaces the worst harmony vector in the HM, if its fitness (judged in terms of the objective function value) is better than the second one. The New Harmony vector is included in the HM and the existing worst harmony vector is excluded from the HM. In step 05, if the stopping criterion (for example, maximum number of improvisations, NI) is satisfied, computation is terminated. Otherwise, Steps 03 and 04 are repeated.

The HMCR and PAR parameters of the GBHS help the method in searching for globally and locally improved solutions, respectively. PAR has a profound effect on the performance of the GBHS algorithm. Thus, fine tuning of this parameter is very important.

B. The K-means algorithm

The K-means algorithm is the simplest and most commonly-used algorithm for clustering employing a Sum of Squared Error (SSE) criterion based on (1).

$$SSE = \sum_{j=1}^{k} \sum_{i=1}^{n} P_{i,j} \left\| x_i - c_j \right\|^2 \quad (1)$$

This algorithm is popular because it finds the local minimum (or maximum) in a search space, it is easy to implement, and its time complexity is O(n). Unfortunately, the quality of the result is dependent on the initial points and may converge to a local minimum of the criterion function value if the initial partition is not properly chosen [13, 35-37]. K-means inputs are: The number of clusters (K value) and a set (table, array or collection) containing n objects (or registers) in a D-dimensionality feature space, formality defined by $X = \{x_1, x_2, .., x_n\}$(In our case, x_i is a row vector, for implementation reasons). K-means outputs are a set containing K centers. The steps in the procedure of K-means can be summarized as shown in Fig 2.

01	Select an Initial Partition (k centers)
	Repeat
02	Re-compute Membership
03	Update Centers
04	Until (Stop Criterion)
05	Return Solution

Fig 2. The K-means algorithm

In step 01, there are several approaches to select K initial centers [38], for example Forgy suggested selecting K instances randomly from the data set and McQueen suggested selecting the first K points in the data set as the preliminary seeds and then using an incremental strategy to update and select the real K centers of the initial solution [38]. In step 02, it is necessary to recompute membership according to the current solution. Several similarity or distance measurements can be used. In this paper, we used Cosine similarity formality defined as (2).

$$Sim(d,q) = \frac{\sum_{i=1}^{D} W_{i,d} \times W_{i,q}}{\sqrt{\sum_{i=1}^{D} W^2_{i,d}} \sqrt{\sum_{i=1}^{D} W^2_{i,q}}} \quad (2)$$

In the literature of partitional clustering, various criteria have been used to compare two or more solutions and decide which one is better [12, 39]. The most popular criteria are based on the within-cluster and between-cluster scatter matrices. In this research, two criteria were used to find the number of clusters automatically. The Bayesian Information Criterion (BIC) [21] expressed by (4) and the Davies–Bouldin (DB) index [29] expressed by (5).

$$BIC = n \times Ln\left(\frac{SSE}{n}\right) + k \times Ln(n) \quad (3)$$

$$DB = \frac{1}{k}\sum_{i=1}^{k} R_i \quad where \quad R_i = \max_{j, j \neq i}\left\{\frac{S_i + S_j}{d_{i,j}}\right\}$$
$$d_{i,j} = 1 - Sim_{\cos}(z_i, z_j)$$
$$S_i = \frac{1}{|C_i|}\sum_{x \in C_i}(1 - Sim_{\cos}(x, z_i)) \quad (5)$$

Where $|C_i|$ is the number of documents in cluster C_i and center of this cluster. S_i is the average cosine dissimilarity be documents in cluster C_i and its centroid z_i. $d_{i,j}$ represents th between cluster C_i and C_j.

III. THE PROPOSED ALGORITHM

The proposed new algorithm, called Iterative Global-Best Harmony Search K-means Algorithm (**IGBHSK**) is a description-centric algorithm [8] for web clustering engines. This algorithm uses the GBHS algorithm as a global strategy of search in the whole solution space, the K-means algorithm as a local strategy for improving solutions; and the Bayesian Information Criterion (BIC) or the Davies-Bouldin index to find the number of clusters automatically. IGBHSK has a main routine that performs six basic steps (see Fig 4):

1: **Initialize algorithm parameters**: In this research, the optimization problem lies in minimizing the BIC or DB criteria, called Fitness function. IGBHSK needs one specific parameter, the Best Memory Results Size (BMRS) and other parameters from the GBHS algorithm: Harmony Memory Size (HMS, a typical value is between 4 and 10), Harmony Memory Considering Rate (HMCR, a typical value is 0.95), Pitch Adjusting Rate (PAR, a typical value is between 0.3 and 0.99), and the Number of Improvisations (NI) or stopping criterion [25, 28, 33, 40].

2: **Document preprocessing**: Initially, Lucene is used at a document pre-processing stage. The pre-processing stage includes: stop word removal, Porter's stemming algorithm and the building of the Term-Document Matrix (TDM with N documents by D dimensions or terms). Optionally, the FP-Growth [5, 23, 41] algorithm is used to build a Frequent Term-Document Matrix (FTDM). FTDM is used to reduce the high dimensionality of document collections. Also, the dimensions with a range equal to zero (0) are removed.

3: **Initialize the best memory results and call the GBHSK routine**: The Best Memory Results (BMR) is a memory location where the best solution vectors are stored (see **Fig 3**). Each row in BMR stores the result of one call to the Global-Best Harmony Search K-means (GBHSK) routine (see section 0), in a basic cycle. Each row vector in BMR has two parts: The Centroids and the Fitness value of that vector.

4: **Select the best result**: Find and select the best result from the Best Memory Results (BMR). The best result is the row with the lowest fitness value (minimize *f(x)*). Then return this row as the best clustering solution (centroids and fitness).

$$BMR = \begin{bmatrix} Centroids_1 & Fitness_1 \\ Centroids_2 & Fitness_2 \\ \vdots & \vdots \\ Centroids_{BMRS-1} & Fitness_{BMRS-1} \\ Centroids_{BMRS} & Fitness_{BMRS} \end{bmatrix}$$

Fig 3. Best Memory Results

5: **Assign labels to clusters**: the IGBHSK algorithm contemplates two methods of assigning labels to each cluster. The first one is a set of Statistically Representative Terms (SRT) based on the probabilistic concept introduced by Smith and Medin [42] and the second one is similar to Lingo [17], based on Frequent PHrases (FPH). See sections 0 and C for more details.

6: **Overlap clusters**: Finally, each cluster includes documents that fall into other clusters too, if these documents are at a distance less than or equal to the average distance of the cluster.

IGBHSK can be executed in different ways (see Fig 5). The first choice to be made is related to the document representation model (TDM or FTDM). Secondly, it is necessary to choose the fitness function (BIC or DB), and finally, it is necessary to choose the labeling method (SRT or FPH).

```
01  Initialize algorithm parameters
02  Document preprocessing
      Stop word removal
      Porter's stemming algorithm
      Term-Document matrix (TDM) or Frequent Term-
       Document matrix (FTDM) building
      Eliminate dimensions with a range equal to zero
03  Initialize the BMR and call the GBHSK routine
      For each i Є [1, BMRS] do
          BMR[i] = GBHSK (TDM or FTDM)
Next-for
04  Select the best result
05  Assign labels to clusters
06  Overlap clusters
```

Fig 4. IGBHSK algorithm

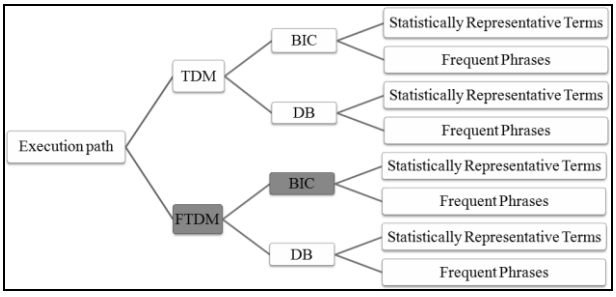


Fig 5. Execution paths of IGBHSK

A. *The GBHSK routine*

In GBHSK routine, each solution vector used has a different number of clusters (centroids), and the objective function (based on BIC or DB) depends on the centroids location in each vector solution and the number of centroids (K value). The GBHSK routine is the GBHS algorithm with some changes, which works as follows (see **Fig 6**):

1: **Initialize the Harmony Memory**: the HM is a memory location where all the solution vectors are stored. Each vector solution is created with a random number of centroids (k < Kmax), initial location of centroids (k centroids with Forgy strategy and values in all dimensions) and Fitness for this solution. The Initial centroids are selected randomly from the original data set (unlike the original GBHS algorithm). Next, **one cycle of the K-means algorithm** (**Fig 2** steps 02 and 03, called **1-means**) is executed and then the fitness value for this solution is calculated. The general structure of HM is similar to BMR. In summary, HMS vector solutions (centroids) are generated and then fitness value for each vector is calculated.

2: **Improvise a New Harmony**: a New Harmony vector (centroids) is generated. A variation of step 3 in the original GBHS algorithm to create centroids in the current solution is used. The random selection process is executed from the original data set (Forgy strategy). Next, 1-means is executed and then the fitness value for this solution is calculated.

3: **Update the Harmony Memory**: the New Harmony vector replaces the worst harmony vector in the HM, if its fitness value is better than the second one.

4: **Check the Stopping Criterion**: if the maximum number of improvisations (NI) is satisfied, the iteration is terminated. Otherwise, Steps 2 and 3 in GBHSK are repeated.

5: **Select the Best Harmony in HM**: the best harmony, which has the minimum fitness value, is found and selected. Then, the K-means algorithm (**Fig 2** without step 01, because this solution has information about initial centroids) is executed and then, a new value of fitness with the final location of centroids is calculated.

6: **Return the Best Result in Harmony Memory**: Return the best harmony (centroids and fitness) to IGBHSK.

B. *Statistically Representative Terms for Labeling*

A cluster is represented by a probabilistic concept, which is a set of statistically representative terms (terms and their associated frequencies in the cluster) [42]. This method works as follows:

1: **Initialize algorithm parameters**: The ***Max Term Threshold*** and the ***Min Frequency Term Threshold*** should be defined. Max Term Threshold represents the maximum number of terms that can comprise the label of each cluster. The Min Frequency Term Threshold represents the percentage of the total sum of the frequencies of all terms to be achieved so that the terms are considered as a label of each cluster.

2: **Building of the "Others" label and cluster**: At the end, if some documents don't match with the label of the cluster,

then they are sent to other clusters. This only occurs when the source is a FTDM matrix.

3: **Candidate label induction**: For each cluster, a new Term-document matrix is created with only the frequency of the cluster's documents. Next, total frequency by terms is calculated and then the terms that reach the Min Frequency Term Threshold are selected. Later, the terms with the highest frequencies are selected, as long as they do not exceed the Max Term Threshold.

4: **Eliminate repeated terms**: To define the final labels, repeated terms in candidate labels are eliminated.

```
01   Initialize the Harmony Memory (HM): define centroids
     (forgy strategy), execute 1-means and Calculate fitness
     (BIC or DB) for each solution vector generated in HM.
02   Improvise a new harmony: define k centroids for this
     solution.
     For i=1 to D (number of dimensions) do
          If U (0, 1) ≤ HMCR then
          Begin  /*memory consideration*/
                    j ~ U (1… HMS); c ~ U (1… HM[j].k)
                    NewCentroid [i] = HM [j].Centroid[c][i]
                    If U(0,1) ≤ PAR then
                    Begin /*pitch adjustment*/
                              c ~ U (1… HM[best].k)
                              NewCentroid [i] = HM[best]
.Centroid[c][i]
                    End-if
          Else /*random selection with forgy strategy*/
                    Rand ~ U (1… N)
                    NewCentroid [i] = TDM[Rand][i] or FTDM
          End-if
     Next-for
     Execute 1-means and Calculate fitness (BIC) for new
     harmony
03   Update the harmony memory: the new harmony vector
     replaces the worst harmony vector in the HM, if its fitness
     value is better than the second one.
04   Check the stopping criterion: if the maximum number of
     improvisations (NI) is satisfied, iteration is terminated.
     Otherwise, Steps 02 and 03 are repeated.
05:  Select the best harmony in HM: find the best harmony,
     execute K-means and Calculate fitness (BIC) for best
     harmony.
06:  Return the best harmony to IGBHSK.
```

Fig 7. Steps in the GBHSK routine

5: **Visual improving of labels**: each term in the label is replaced with the longest representation of it, but if the term is a verb, this term is replaced by the infinitive of the verb.

C. *Frequent Phrases for Labeling*

This step corresponds to step 2 "Frequent Phrase Extraction" in Lingo [18], but in WDC-NMA this method is used for each generated cluster in previous steps. By the above method, some changes were made to the original algorithm, so that it works as follows:

1: **Conversion of the representation**: Each document in the current cluster is converted from character-based to word-based representation.

2: **Document concatenation**: All documents in the current cluster are concatenated and a new document with the inverted version of the concatenated documents is created.

3: **Complete phrase discovery**: Right-complete phrases and left-complete phrases are discovered in the current cluster, then the right-complete phrases and left-complete phrases are alphabetically sorted, and then the left- and right-complete phrases are combined into a set of complete phrases.

4: **Final selection**: Terms and phrases whose frequency exceeds the ***Term Frequency Threshold*** are selected for the current cluster.

5: **Building of the "Others" label and cluster**: if some documents don't reach the Term Frequency Threshold, then they are sent to the other clusters.

6: **Cluster label induction**: In the current cluster, a Term-document matrix is built. Then, using cosine similarity, the best candidate terms or phrases for the cluster (which optimize SSE) is selected.

D. *Complexity*

IGBHSK repeats the GBHSK routine BMRS times and then makes the sorting of a vector with BMRS rows. The major computational load occurs in each step of the GBHSK routine. GBHSK routine generates HMS solution vectors and then NI improvisations. For each vector solution generated in GBHSK routine, one step of the K-means algorithm is executed and fitness value (BIC or DB Index) is calculated. Finally, GBHSK routine finds and selects the best solution, performs the K-means algorithm for this solution and re-calculates the fitness value. One step of the K-means algorithm and the fitness value calculating of a given solution take $O(n*k*D)$ and $O(n*D)$ times, respectively. The total K-means algorithm and to re-calculate the fitness value take $O(n*k*D*L)$ times (where L is the number of iterations taken by the K-means algorithm to converge). Furthermore the labeling of groups takes O(n*D) times. Perform the overlap of the groups takes O(k*n*D) times. Therefore, the overall complexity of the proposed algorithm is $O(n*k*D*(L+HMS+NI)*BMRS)$.

IV. THE PROPOSED ALGORITHM

A. *Data Sets*

The proposed new algorithm was used for text document clustering on the Reuters-21578 corpus, which is one of the most-widely adopted benchmark datasets in the text mining field. Four data sets were used, namely: Data set 1 (DS1) with 200 documents, 4 topics (money-supply, coffee, sugar, and interest), 2613 terms in the vocabulary and 981 frequent terms. Each topic has between 40 and 60 documents. Data set 2 (DS2) with 400 documents from 5 topics (money-supply, coffee, sugar, interest, and ship), 3991 terms in the vocabulary

and 189 frequent terms. Each topic has between 60 and 100 documents. Data set 3 (DS3) with 200 documents from 4 topics (money-supply, coffee, sugar, and interest), 2697 terms in the vocabulary and 419 frequent terms. Each topic has exactly 50 documents. Data set 4 (DS4) with 400 documents from 5 topics (money-supply, coffee, sugar, interest, and ship), 3977 terms in the vocabulary and 203 frequent terms. Each topic has exactly 80 documents.

The algorithm was also used for text document clustering on document extracts from DMOZ (Open Directory Project). Four data sets were used: Data set 5 (DS5) with 112 documents, 3 topics (Coins, Diamonds, and Motorcycles), 350 terms in the vocabulary and 63 frequent terms. Each topic has between 27 and 43 documents; Data set 6 (DS6) with 112 documents, 4 topics (Diamonds, Food, Greek, and Virtual Reality), 457 terms in the vocabulary and 39 frequent terms. Each topic has between 13 and 43 documents; Data set 7 (DS7) with 135 documents, 4 topics (Coins, Diamonds, Travel and Tourism and Weather), 425 terms in the vocabulary and 63 frequent terms. Each topic has between 10 and 43 documents; and Data set 8 (DS8) with 133 documents, 5 topics (Algorithms, Number Theory, Volleyball, Adventure, and Dogs), 589 terms in the vocabulary and 24 frequent terms. Each topic has between 11 and 42 documents.

Since the clustering of web documents working with snippets and an average of 100-300 documents, test datasets used are appropriated for the objectives of the algorithm.

B. IGBHSK Parameters and Measures

Most of the parameter values were equal for all data sets. BRMS equal to 5, HMS equal to 5, HMCR equal to 0.95, PAR equal to 0.35, and NI equal to 30. Kmax value was equal to $\sqrt{N/2}+1$, where N is the number of documents. For FP-Growth algorithm a support value of 10% was used for Reuter's data sets and a support value of 5% was used for the DMOZ data sets.

Fig 7 shows the lineal relation between the execution time and the number of improvisations (NI). If NI is equal to 30, n (number of documents in web document clustering) is around 200, D is around 3000 terms, BRMS equal to 5, HMS equal to 5, and k is around 10, then the execution time of IGBHSK is around 0.35 seconds approximately.

C. Results

Two main questions were formulated: is the number of clusters correctly identified? And, are the documents grouped in an appropriate way? To answer these questions, Precision (P), Recall (R) and F-measure (F) or balanced F-score were computed, since the "true" clusters of each data set are known [44]. High values of P, R, and F are desirable.

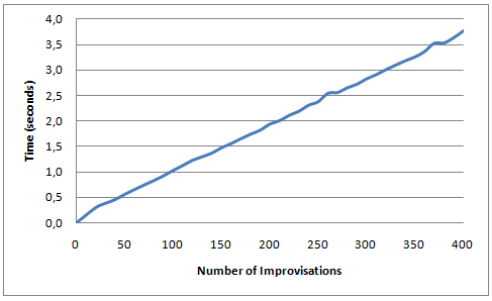


Fig 8. Time (seconds) Vs Number of Improvisations

The algorithm was run 10 times over the Term-Document Matrix (TDM) with BIC criterion and the averages to show them as results were calculated. Next, the same algorithm was run 10 times using DB index over the same matrix (TDM). These promising results are shown in Table I.

TABLE I
PRECISION (P), RECALL(R) AND F-MEASURE (F) BY TWO CRITERIA (BIC AND DB) IN TERM-DOCUMENT MATRIX (TDM)

	K			P		R		F	
	Ideal	**BIC**	**DB**	**BIC**	**DB**	**BIC**	**DB**	**BIC**	**DB**
DS 1	4	**9,3**	9,5	**84,0%**	79,4%	51,6%	**56,0%**	63,8%	**65,5%**
DS 2	5	13,1	**12,2**	**82,6%**	77,0%	48,3%	**56,9%**	60,5%	**65,3%**
DS 3	4	8,7	**8,5**	**83,4%**	75,5%	50,0%	**56,1%**	62,0%	**64,0%**
DS 4	5	13,3	**12,3**	**86,3%**	76,4%	48,5%	**54,3%**	61,8%	**63,1%**
DS 5	3	6,6	**4,1**	**97,4%**	90,8%	65,7%	**84,6%**	78,3%	**87,4%**
DS 6	4	7	**3,1**	**85,6%**	49,8%	**62,6%**	54,0%	**72,0%**	51,3%
DS 7	4	7,4	**2,6**	**91,3%**	49,9%	**71,2%**	59,3%	**79,7%**	53,8%
DS 8	5	7,6	**6,9**	**66,9%**	59,8%	**50,5%**	49,6%	**57,3%**	53,9%
Avg.	4,3	9,1	**7,4**	**84,7%**	69,8%	56,0%	**58,8%**	**66,9%**	63,1%

Later, the algorithm was run 10 times over the Frequent Term-Document Matrix (FTDM) with BIC criterion and the averages to show them as results were calculated. Next, the same algorithm was run 10 times using DB index over the same matrix (FTDM). These promising results are shown in Table II.

In general, the algorithm obtained better precision results using the BIC criterion over the Frequent Term-Document Matrix, but DB index reported more appropriate values of k (number of clusters) in FTDM. In relation to F-measure, results are better for the Term-Document Matrix (5% more), but in this execution path of IGBHSK, there is no dimensionality reduction.

D. Comparison

Carrot2 is a web search that performs web document clustering to show the results of queries. Carrot2 implements the Lingo model. It is based on Singular Value Decomposition and frequent phrases. To compare IGBHSK (FTDM-BIC-FPH) with Carrot2, the last version of Carrot2 Workbench was used.

TABLE II
PRECISION (P), RECALL(R) AND F-MEASURE (F) BY TWO CRITERIA IN FREQUENT TERM-DOCUMENT MATRIX (FTDM)

	K	P			R		F	
	Ideal	**BIC**	**DB**	**BIC**	**DB**	**BIC**	**DB**	**BIC**	**DB**
DS 1	4	10	**2,3**	**86,1%**	37,6%	**48,9%**	46,5%	**62,3%**	41,3%
DS 2	5	14,2	**12,3**	**88,3%**	85,5%	42,9%	**48,6%**	57,7%	**61,8%**
DS 3	4	10,2	**8,1**	**85,7%**	78,3%	49,7%	**53,2%**	**62,7%**	62,7%
DS 4	5	14,2	**8,7**	**89,0%**	72,4%	45,0%	**51,3%**	**59,7%**	58,4%
DS 5	3	7,5	**4,5**	**90,8%**	68,6%	47,9%	**56,8%**	**62,6%**	61,2%
DS 6	4	7,6	**4,2**	**80,4%**	51,0%	**50,6%**	47,9%	**62,0%**	48,8%
DS 7	4	8,8	**4,8**	**87,9%**	55,6%	**54,7%**	49,2%	**67,4%**	50,9%
DS 8	5	8,8	**4,6**	**73,1%**	40,4%	**51,7%**	35,7%	**60,4%**	36,6%
Avg.	4,3	10,2	**6,2**	**85,2%**	61,2%	**48,9%**	48,6%	**61,9%**	52,7%

This version of Carrot2 is a desktop application and it can be executed with different sources (not only web search results). Data sets based on DMOZ (DS 5 to 8) were used to compare IGBHSK with Carrot2. Table III shows the results of Precision, Recall, F-measure, Number of Labels that were really Representative of each cluster (NRL) and the number of documents in the Other Topics Label (OTC). In general, IGBHSK had better results (precision, recall and F-measure) than Carrot2. IGBHSK defined a better value of k (number of clusters) than Carrot2. IGBHSK had a higher proportion of representative labels than Carrot2, and finally, Carrot2 reported a lot of documents in the other topics cluster (bad for the end user). **Fig 8** shows 9 clusters and their respective label for DS 8 with 5 original topics (Algorithms, Number Theory, Volleyball, Adventure, and Dogs) while **Fig 9** shows 11 clusters generated by Carrot2. Note that the "other topics" cluster has 88 documents.

TABLE III
PRECISION (P), RECALL(R), F-MEASURE (F), NUMBER O REPRESENTATIVE LABELS (NRL) AND NUMBER OF DOCUMENTS IN OTHER TOPICS CLUSTER (OTC) BY IGBHSK AND CARROT2

		DS 5	**DS 6**	**DS 7**	**DS 8**
K	Real	3	4	4	5
	Carrot2	9	9	11	11
	IGBHSK	**7,5**	**7,6**	**8,8**	**8,8**
P	Carrot2	78.2%	68.1%	59.7%	58.5%
	IGBHSK	**90,8%**	**80,4%**	**87,9%**	**73,1%**
R	Carrot2	33.5%	42.8%	26.0%	29.4%
	IGBHSK	**47,9%**	**50,6%**	**54,7%**	**51,7%**
F	Carrot2	46.9%	52.6%	36.2%	39.2%
	IGBHS	**62,9%**	**62,0%**	**67,4%**	**60,4%**

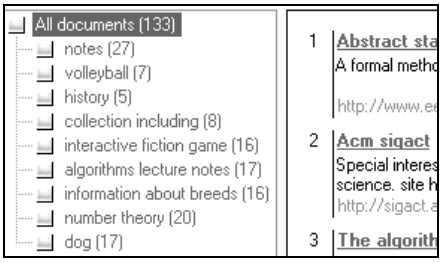


Fig 9. Precision for BIC and DB criteria

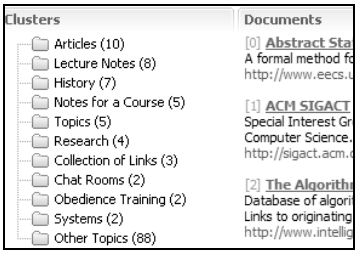


Fig 10. Results of Data set 8 in Carrot2

E. User evaluation

A user-based evaluation method was used to assess the clustering results produced by IGBHSK when data sources are Google, Yahoo! and Bing. Users were divided into 4 groups for a total of 62 people. For each cluster, the user answered whether or not:

- (Q1) the cluster label is in general representative of the cluster (much – R3, little – R2, or nothing - R1), and
- (Q2) the cluster is useful (R3), moderately useful (R2) or useless (R1).

Then, for each document in each cluster, the user answered whether or not:

- (Q3) the document matches with the cluster (very well matching – R3, moderately matching – R2, or not-matching – R1), and
- (Q4) the document relevance (location) in the cluster was adequate (adequate – R3, moderately suitable – R2, or inadequate – R1).

A standard information retrieval metrics -precision and recall- to assess all user responses was used in each cluster. General results of IGBHSK (FTDM-BIC-FPH) are shown in **Fig 10.** Most user responses are R3 or R2. Therefore, results are very promising and it is necessary to do a set of very controlled experiments with users, in order to generalize results.

For question 1 (representative cluster labels), general results are very good (R3) and best results are obtained when the algorithm uses term-document matrix and statistically represented terms to define the cluster's label. In this case the fitness function was not very important (see **Fig 11**).

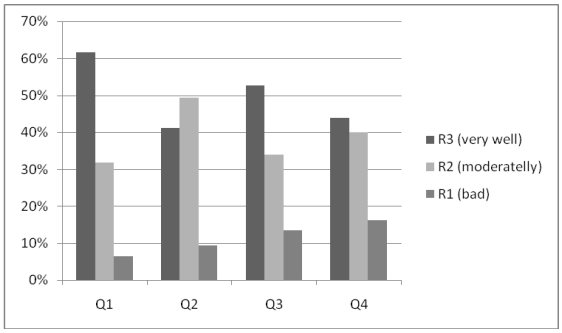


Fig 11. General results for the four questions

For question 2 (useful cluster), general results are good (between R3 and R2) and similarly for Q1, best results are obtained when the algorithm uses term-document matrix and statistically represented terms to define the cluster's label. In this case the fitness function was not very important (see **Fig 12**). For question 3 (document matching with the cluster), general results are good (between R3 and R2) and best results are obtained when the algorithm uses frequent term-document matrix and Bayesian information criterion. In this case the labeling method was not very important. For question 4 (document relevance in cluster), general results are good (between R3 and R2) and similarly for Q3, best results are obtained when the algorithm uses frequent term-document matrix and Bayesian information criterion. In this case the labeling method was not very important. Worst results in all questions occur when Davies- Bouldin index and frequent phrases was used. In this case, the document representation model was not very important.

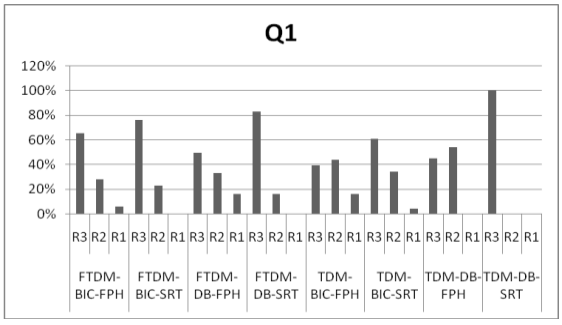


Fig 12. Specific results for Q1 in different sceneries

I. CONCLUSIONS AND FUTURE WORK

The IGBHSK algorithm has been designed, implemented and evaluated. IGBHSK is a web document clustering algorithm based on the hybridization of the Global-Best Harmony Search (global search strategy) and the K-means algorithm (local solution improvement strategy) with the capacity of automatically defining the number of clusters.

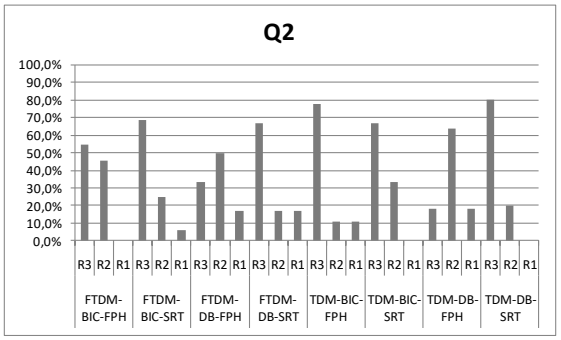


Fig 13. Specific results for Q2 in different sceneries

IGBHSK shows promising experimental results (in standard datasets and user-based evaluation). IGBHSK shows better results when using Frequent Term-Document matrix and the Bayesian Information Criterion. The overall complexity of IGBHSK is *O(n*k*D*(L+HMS+NI)*BMRS)*, so IGBHSK can be used with large data sets in off line text clustering task. Frequent terms sets (based on FP-Growth algorithm) can be used with IGBHSK to reduce the high dimensionality of the vocabulary and to improve precision in results.

There are several tasks for future work. Among them: applying the IGBHSK algorithm to several synthetic data sets (Text Retrieval Conference-TREC, other data sets based on Reuters-21578, High Accuracy Retrieval from Documents – HARD, Track of Text Retrieval Conference, among others) and real data sets (other data sets based on DMOZ, Google results, Yahoo results, among others); comparing the new algorithm with other traditional and evolutionary algorithms; making use of WordNet [43] to work with concepts instead of terms and comparing the results.

II. ACKNOWLEDGMENTS

The work in this paper was supported by a Research Grant from the University of Cauca under Project VRI-2560 and the National University of Colombia.

III. REFERENCES

[1] R. Baeza-Yates, A. and B. Ribeiro-Neto, *Modern Information Retrieval*: Addison-Wesley Longman Publishing Co., Inc., 1999.
[2] G. Mecca, S. Raunich, and A. Pappalardo, "A new algorithm for clustering search results," *Data & Knowledge Engineering*, vol. 62, pp. 504-522, 2007.
[3] C. Claudio, O. Stanislaw, ski, R. Giovanni, and W. Dawid, "A survey of Web clustering engines," *ACM Comput. Surv.*, vol. 41, pp. 1-38, 2009.
[4] Z. Oren and E. Oren, "Web document clustering: a feasibility demonstration," in *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval* Melbourne, Australia: ACM, 1998.
[5] L. Jing, "Survey of Text Clustering," 2006.
[6] K. Hammouda, "Web Mining: Clustering Web Documents A Preliminary Review," 2001.

[7] A. K. Jain and R. C. Dubes, *Algorithms for clustering data*: Prentice-Hall, Inc., 1988.
[8] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: a review," *ACM Comput. Surv.*, vol. 31, pp. 264-323, 1999.
[9] M. Steinbach, G. Karypis, and V. Kumar, "A comparison of document clustering techniques," 2000.
[10] Y. Li, S. M. Chung, and J. D. Holt, "Text document clustering based on frequent word meaning sequences," *Data & Knowledge Engineering*, vol. 64, pp. 381-404, 2008.
[11] D. Zhang and Y. Dong, "Semantic, Hierarchical, Online Clustering of Web Search Results," in *Advanced Web Technologies and Applications*, 2004, pp. 69-78.
[12] S. Osiński, "An Algorithm for clustering of web search results." vol. Master Poland: Poznań University of Technology,, 2003, p. 91.
[13] S. Osiński, J. Stefanowski, and D. Weiss, "Lingo: Search results clustering algorithm based on Singular Value Decomposition," 2004.
[14] S. Osiński and D. Weiss, "Conceptual clustering using Lingo algorithm: Evaluation on Open Directory Project data," 2004.
[15] S. Osiński and D. Weiss, "A concept-driven algorithm for clustering search results," *Intelligent Systems, IEEE*, vol. 20, pp. 48-54, 2005.
[16] S. Osiński and D. Weiss, "Carrot 2: Design of a Flexible and Efficient Web Information Retrieval Framework," in *Advances in Web Intelligence*, 2005, pp. 439-444.
[17] S. Osiński, "Improving quality of search results clustering with approximate matrix factorizations," in *28th European Conference on IR Research (ECIR 2006)*, London, UK, 2006, pp. 167,178.
[18] F. Beil, M. Ester, and X. Xu, "Frequent term-based text clustering," in *KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, Edmonton, Alberta, Canada, 2002, pp. 436-442.
[19] B. Fung, K. Wang, and M. Ester, "Hierarchical document clustering using frequent itemsets," in *Proceedings of the SIAM International Conference on Data Mining*, 2003.
[20] Z. Geem, J. Kim, and G. V. Loganathan, "A New Heuristic Optimization Algorithm: Harmony Search," *SIMULATION*, vol. 76, pp. 60-68, 2001.
[21] R. Forsati, M. R. Meybodi, M. Mahdavi, and A. G. Neiat, "Hybridization of K-Means and Harmony Search Methods for Web Page Clustering," in *Web Intelligence and Intelligent Agent Technology, 2008. WI-IAT '08. IEEE/WIC/ACM International Conference on*, 2008, pp. 329-335.
[22] M. Mahdavi and H. Abolhassani, "Harmony K-means algorithm for document clustering," *Data Mining and Knowledge Discovery*, vol. 18, pp. 370-391, 2009.
[23] M. Mahdavi, M. H. Chehreghani, H. Abolhassani, and R. Forsati, "Novel meta-heuristic algorithms for clustering web documents," *Applied Mathematics and Computation*, vol. 201, pp. 441-451, 2008.
[24] W. Song, C. H. Li, and S. C. Park, "Genetic algorithm for text clustering using ontology and evaluating the validity of various semantic similarity measures," *Expert Systems with Applications*, vol. 36, pp. 9095-9104, 2009.
[25] W. Song and S. Park, "Genetic Algorithm-Based Text Clustering Technique," in *Advances in Natural Computation*, 2006, pp. 779-782.
[26] K. Lee and Z. Geem, "A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice," *Computer Methods in Applied Mechanics and Engineering*, vol. 194, pp. 3902-3933, 2005.
[27] L. d. Santos Coelho and D. L. de Andrade Bernert, "An improved harmony search algorithm for synchronization of discrete-time chaotic systems," *Chaos, Solitons & Fractals*, vol. In Press, Corrected Proof, 2008.
[28] M. G. H. Omran and M. Mahdavi, "Global-best harmony search," *Applied Mathematics and Computation*, vol. 198, pp. 643-656, 2008.
[29] J. Kennedy and R. C. Eberhart, "Particle Swarm Optimization," in *IEEE Int'l. Conf. on Neural Networks.*, Perth, Australia, 1995, pp. 1942–1948.
[30] P. Berkhin, "Survey Of Clustering Data Mining Techniques," 2002.
[31] J. Han, M. Kamber, and A. K. H. Tung, "Spatial Clustering Methods in Data Mining: A Survey," in *Geographic Data Mining and Knowledge Discovery*: Taylor and Francis, 2001.
[32] G. H. O. Mahamed, P. E. Andries, and S. Ayed, "An overview of clustering methods," *Intell. Data Anal.*, vol. 11, pp. 583-605, 2007.
[33] S. J. Redmond and C. Heneghan, "A method for initialising the K-means clustering algorithm using kd-trees," *Pattern Recognition Letters*, vol. 28, pp. 965-973, 2007.
[34] A. Webb, *Statistical Pattern Recognition, 2nd Edition*: {John Wiley & Sons}, 2002.
[35] M. Mahdavi, M. Fesanghary, and E. Damangir, "An improved harmony search algorithm for solving optimization problems," *Applied Mathematics and Computation*, vol. 188, pp. 1567-1579, 2007.
[36] X. Liu and P. He, "A Study on Text Clustering Algorithms Based on Frequent Term Sets," in *Advanced Data Mining and Applications*, 2005, pp. 347-354.
[37] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*, 2nd ed.: Morgan Kaufman Publishers, 2006.
[38] H. Ralambondrainy, "A conceptual version of the K-means algorithm," *Pattern Recognition Letters*, vol. 16, pp. 1147-1157, 1995.
[39] C. Fellbaum, *WordNet: An Electronic Lexical Database*: MIT Press, 1998.

REFERENCIAS

[1] M. Porter, "An algorithm for suffix stripping," *Program,* vol. 14, pp. 130-137, 1980.

[2] Han J., Pei J., and Yin Y., "Mining Frequent Patterns without Candidate Generation," *ACM SIGMOD Record,* vol. 29, 2000.

[3] Naranjo C. Roberto C. and Sierra M. Luz Marina, "Herramienta software para el análisis de canasta de mercado sin selección de candidatos," *REVISTA INGENIERÍA E INVESTIGACIÓN,* vol. 29, 2009.

[4] Borgelt C., "Frequent Pattern Mining," *Department of Knowledge Processing and Language Engineering -School of Computer Science,* 2005.

[5] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*, 2nd ed.: Morgan Kaufman Publishers, 2006.

[6] R. Baeza-Yates, A. and B. Ribeiro-Neto, *Modern Information Retrieval*: Addison-Wesley Longman Publishing Co., Inc., 1999.

[7] G. Mecca, S. Raunich, and A. Pappalardo, "A new algorithm for clustering search results," *Data & Knowledge Engineering,* vol. 62, pp. 504-522, 2007.

[8] C. Claudio, O. Stanislaw, ski, R. Giovanni, and W. Dawid, "A survey of Web clustering engines," *ACM Comput. Surv.,* vol. 41, pp. 1-38, 2009.

[9] Z. Oren and E. Oren, "Web document clustering: a feasibility demonstration," in *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval* Melbourne, Australia: ACM, 1998.

[10] L. Jing, "Survey of Text Clustering," 2006.

[11] K. Hammouda, "Web Mining: Clustering Web Documents A Preliminary Review," 2001.

[12] A. K. Jain and R. C. Dubes, *Algorithms for clustering data*: Prentice-Hall, Inc., 1988.

[13] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: a review," *ACM Comput. Surv.,* vol. 31, pp. 264-323, 1999.

[14] M. Steinbach, G. Karypis, and V. Kumar, "A comparison of document clustering techniques," 2000.

[15] Y. Li, S. M. Chung, and J. D. Holt, "Text document clustering based on frequent word meaning sequences," *Data & Knowledge Engineering,* vol. 64, pp. 381-404, 2008.

[16] D. Zhang and Y. Dong, "Semantic, Hierarchical, Online Clustering of Web Search Results," in *Advanced Web Technologies and Applications*, 2004, pp. 69-78.

[17] S. Osiński, "An Algorithm for clustering of web search results." vol. Master Poland: Poznań University of Technology,, 2003, p. 91.

[18] S. Osiński, J. Stefanowski, and D. Weiss, "Lingo: Search results clustering algorithm based on Singular Value Decomposition," 2004.

[19] S. Osiński and D. Weiss, "Conceptual clustering using Lingo algorithm: Evaluation on Open Directory Project data," 2004.

[20] S. Osiński and D. Weiss, "A concept-driven algorithm for clustering search results," *Intelligent Systems, IEEE,* vol. 20, pp. 48-54, 2005.

[21] S. Osiński and D. Weiss, "Carrot 2: Design of a Flexible and Efficient Web Information Retrieval Framework," in *Advances in Web Intelligence*, 2005, pp. 439-444.

[22] S. Osiński, "Improving quality of search results clustering with approximate matrix factorizations," in *28th European Conference on IR Research (ECIR 2006)*, London, UK, 2006, pp. 167,178.

[23] F. Beil, M. Ester, and X. Xu, "Frequent term-based text clustering," in *KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, Edmonton, Alberta, Canada, 2002, pp. 436-442.

[24] B. Fung, K. Wang, and M. Ester, "Hierarchical document clustering using frequent itemsets," in *Proceedings of the SIAM International Conference on Data Mining*, 2003.

[25] Z. Geem, J. Kim, and G. V. Loganathan, "A New Heuristic Optimization Algorithm: Harmony Search," *SIMULATION,* vol. 76, pp. 60-68, 2001.

[26] R. Forsati, M. R. Meybodi, M. Mahdavi, and A. G. Neiat, "Hybridization of K-Means and Harmony Search Methods for Web Page Clustering," in *Web Intelligence and Intelligent Agent Technology, 2008. WI-IAT '08. IEEE/WIC/ACM International Conference on*, 2008, pp. 329-335.

[27] M. Mahdavi and H. Abolhassani, "Harmony K-means algorithm for document clustering," *Data Mining and Knowledge Discovery,* vol. 18, pp. 370-391, 2009.

[28] M. Mahdavi, M. H. Chehreghani, H. Abolhassani, and R. Forsati, "Novel meta-heuristic algorithms for clustering web documents," *Applied Mathematics and Computation,* vol. 201, pp. 441-451, 2008.

[29] W. Song, C. H. Li, and S. C. Park, "Genetic algorithm for text clustering using ontology and evaluating the validity of various semantic similarity measures," *Expert Systems with Applications,* vol. 36, pp. 9095-9104, 2009.

[30] W. Song and S. Park, "Genetic Algorithm-Based Text Clustering Technique," in *Advances in Natural Computation*, 2006, pp. 779-782.

[31] K. Lee and Z. Geem, "A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice," *Computer Methods in Applied Mechanics and Engineering,* vol. 194, pp. 3902-3933, 2005.

[32] L. d. Santos Coelho and D. L. de Andrade Bernert, "An improved harmony search algorithm for synchronization of discrete-time chaotic systems," *Chaos, Solitons & Fractals,* vol. In Press, Corrected Proof, 2008.

[33] M. G. H. Omran and M. Mahdavi, "Global-best harmony search," *Applied Mathematics and Computation,* vol. 198, pp. 643-656, 2008.

[34] J. Kennedy and R. C. Eberhart, "Particle Swarm Optimization," in *IEEE Int'l. Conf. on Neural Networks.*, Perth, Australia, 1995, pp. 1942–1948.

[35] P. Berkhin, "Survey Of Clustering Data Mining Techniques," 2002.

[36] J. Han, M. Kamber, and A. K. H. Tung, "Spatial Clustering Methods in Data Mining: A Survey," in *Geographic Data Mining and Knowledge Discovery*: Taylor and Francis, 2001.

[37] G. H. O. Mahamed, P. E. Andries, and S. Ayed, "An overview of clustering methods," *Intell. Data Anal.,* vol. 11, pp. 583-605, 2007.

[38] S. J. Redmond and C. Heneghan, "A method for initialising the K-means clustering algorithm using kd-trees," *Pattern Recognition Letters,* vol. 28, pp. 965-973, 2007.

[39] A. Webb, *Statistical Pattern Recognition, 2nd Edition*: {John Wiley & Sons}, 2002.

[40] M. Mahdavi, M. Fesanghary, and E. Damangir, "An improved harmony search algorithm for solving optimization problems," *Applied Mathematics and Computation,* vol. 188, pp. 1567-1579, 2007.

[41] X. Liu and P. He, "A Study on Text Clustering Algorithms Based on Frequent Term Sets," in *Advanced Data Mining and Applications*, 2005, pp. 347-354.

[42] H. Ralambondrainy, "A conceptual version of the K-means algorithm," *Pattern Recognition Letters,* vol. 16, pp. 1147-1157, 1995.

[43] C. Fellbaum, *WordNet: An Electronic Lexical Database*: MIT Press, 1998.