Índex

1. Introducció, motivacions, propòsit i objectius

La confecció d'horaris de institut es un problema que amaga una alta combinatòria, dificultant-ne molt la seva elaboració manual posat que s'han de prendre moltíssimes decisions a cegues, fent que sigui molt probable cometre errors en la confecció. El HSTT (High School TimeTabling) consisteix en la solució de forma automàtica de aquest problema d'alta complexitat (NP). Així doncs el problema consisteix en la configuració automàtica de horaris de institut partint de una sèrie de recursos (per exemple: aules, professors, assignatures, grups) i repartir-los de manera que sigui viable i tenint en compte de manera total o parcial les preferències del professorat en quant a horaris, continuïtat, grups, etc. Tot aixó fa que el problema sigui molt difícil de resoldre, degut al gran nombre de combinacions possibles entre els diferents recursos.

Afegint dificultat al problema, depenguen del país del qual estiguem parlant existeixen una gran diversitat de requisits propis, degut a les característiques pròpies del sistema d'estudis secundaris de cada lloc.

Els objectius d'aquest treball són aprofundir en el problema treballat, l'estat actual d'aquest, estudiar les tècniques més utilitzades per resoldre'l actualment. També es pretén solucionar-lo usant les eines desenvolupades recentment pel grup de recerca de Lògica i Programació. Així s'implementarà un generador d'horaris capaç de resoldre el problema utilitzant aquestes tècniques basades en SMT i SAT.

1.1. Grup de recerca Lògica i Programació

Aquest treball s'emmarca dins del grup de recerca de Lògica i Programació de l'àmbit d'àrea tècnica de la Universitat de Girona.

El grup basa la seva recerca en l'estudi de satisfactibilitat de formules proposicionals booleanes (SAT) i Satisfiability Modulo Theories (SMT) i les seves aplicació per a la resolució de problemes combinatoris com ara problemes de *scheduling* i *planning* arribant a utilitzar amb èxit tècniques innovadores en altres àmbits com pot ser: els problemes de *scheduling* i *planning*.

Durant la elaboració del treball he rebut ajuda i assessorament dels membres del grup, incloent el meu tutor de projecte, el Dr. Josep Suy, qui també en forma part.

2. Estudi de viabilitat

3. Metodologia i Planificació

3.1. Metodologia

La part més important i grossa d'aquest treball és la creació d'un generador automàtic d'horaris utilitzant les eines oferides per el grup de recerca. Primer caldrà estudiar el problema (HSTT) i la seva duresa, per poder ser capaç d'entendre el què estem treballant. Des de aquí es procedirà a la implementació del programa, utilitzant la API SMT creada per el Dr. Jordi Coll, el qual es membre del grup de recerca de Lògica i Programació del departament de Informàtica, Matemàtica Aplicada i Estadística. Aquesta API ens estalviarà la codificació de les restriccions de cardinalitat i les restriccions pseudo-booleanes, apart de oferir-nos una interfície senzilla per poder implementar el model en diferents encodings i múltiples opcions.

3.2. Planificació

Com s'ha dit en l'apartat anterior primer caldrà estudiar el problema HSTT i la seva duresa. Després es procedirà amb el disseny i la implementació del generador. Primer caldrà dissenyar implementar i testejar un *parser* pels fitxers. En aquest pas també cal pensar i implementar en quina estructura es guardaràn aquestes dades i com es transferiràn en el model que es codificarà posteriorment.

Al tenir el *parser* i l'estructura de dades enllestits caldrà començar a estudiar com funciona la API per C++ del Yices i posteriorment la API SMT del Dr. Jordi Coll. Això ens permetrà començar a dissenyar, codificar i testejar el model, que és el següent pas del treball. Al tenir enllestit el model, es faràn les proves de rendiment amb diferents límits de optimització i diferents encodings de les restriccions de cardinalitat. Finalment s'implementarà una forma maca i llegible de mostrar els horaris generats.

Amb això el generador es donarà per acabat i es passarà a la confecció de la memòria del treball.

4. Marc de treball i conceptes previs

4.1. Marc de treball

Com s'ha mencionat en la introducció aquest treball s'enmarca dins del grup de recerca de Lògica i Programació (LAP) del departament de Informàtica, Matemàtica Aplicada i Estadística (IMAE) de la Universitat de Girona. En el treball s'utilitzaràn les eines desenvolupades recentment en el grup de recerca, per ser exactes, la API SMT feta en C++ per el Dr. Jordi Coll. Aquesta API permet codificar problemes SAT, MaxSAT i SMT per a diferents solvers de forma transparent a aquest i te implementades les diferents restriccions de cardinalitat i pseudo-booleanes en les diferents possibles codificacions. També inclou diferents algoritmes d'optimització implementats.

La resta del treball prévi seria el treball també sobre confecció d'horaris fet el 2015 per en Cristòfor Nogueira[1]

4.2. Definició del problema

El problema que es treballa és el de la confecció d'horaris per a institut (HSTT de *High School Time Tables*). Aquest consisteix en assignar a cada assignatura que es fa en un centre l'espai de temps en que s'impartirà i el conjunt de recursos que utilitzarà. Els recursos normalment seràn professors i aules, però es contemplen altres possibles necessitats especials de cada centre, per això es generalitza.

La duresa de aquest problema es troba en assignar un espai de temps per a cada assignatura donant-li els recursos que necessita sense violar cap restricció que aquests tinguin, com podria ser no fer dues assignatures a la vegada en la mateixa aula, o sigui, que al realitzar-se la assignatura tots els seus recursos estiguin disponibles. De la mateixa manera es poden imposar diverses restriccions de naturaleses diferents i amb cada una s'aniràn reduint les possibles combinacions vàlides i fent més i més difícil la generació del horari.

4.2.1 Estudi de duresa

4.2.1.1 Incís en la teoría de la computació

Abans de procedir a l'estudi de la duresa del problema HSTT, caldrà explicar els següents conceptes de la teoria de la computació:

Problemes decidibles i indecidibles Un problema decidible és aquell per el qual existeix una màquina de Turing que para en totes les entrades possibles amb una resposta: sí o no. Aquests problemes també són coneguts com a Turing Decidibles. Així doncs, un problema decidible és aquell pel qual sempre podrem construir un algorisme que sempre respon el problema.

Un problema pot ser semi-decidible, això passa quan una màquina de turing quan l'entrada és acceptada, però es pot penjar o es pot parar quan l'entrada es rebutja. Aquests problemes també son referits com a Turing Reconeixibles.

Un problema indecidible és aquell pel qual no podem construir un algorisme que resolgui el problema en temps finit. Aquests problemes poden ser parcialment decidibles, però sempre hi haurà una condició que portara la màquina de Turing a bucle infinit.

P, NP i NP-Completesa

- **P:** És el conjunt de problems que poden ser resolts en temps polinòmic amb una màquina de Turing determinista.
- **NP:** És el conjunt de problemes que poden ser resolts en temps polinòmic amb una màquina de Turing no determinista.
- **NP-Complet:** És el conjunt de problemes més durs en el conjunt NP. Un problema C és NP-Complet si C és NP i tot problema NP és reduïble a C.
- **NP-Hard:** És el conjunt de problemes als quals es pot reduïr tot problema NP. O sigui, un problema C és NP-Hard si tot problema NP és reduïble a C.
- **Reducció:** Si tenim dos problemes L_1 i L_2 i tenim un algoritme A_2 que resol L_2. Reduir L_1 a L_2 és transformar el problema L_1 a L_2 així poder utilitzar el algoritme A_2 per resoldre el problema, creant així un algoritme A_1 amb l'estructura que es pot veure en la figura 4.1

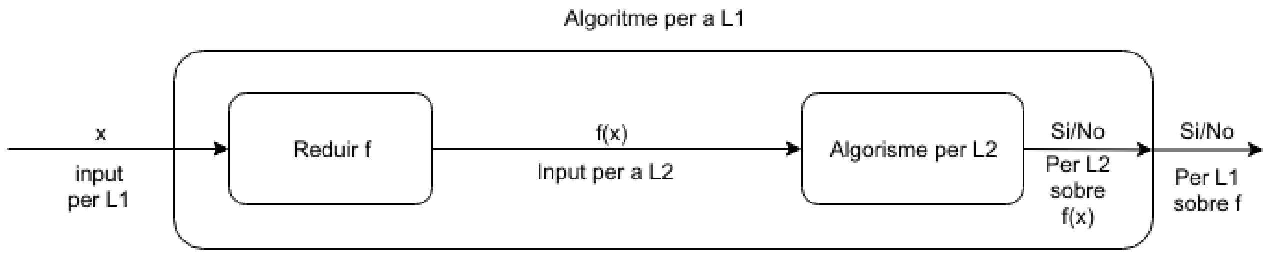

Figura 4.1: Esquema de Reducció

4.2.1.2 Duresa de HSTT

El problema HSTT és clarament decidible i, posat que comprovar si una solució satisfà totes les restriccions imposades és d'ordre polinòmic, aquest pertany al conjunt de problemes NP.

NP-Completesa de HSTT

En aquest apartat s'inclou la demostració que es veu en el treball d'en Cristòfor Nogueira[1].

Es pot demostrar la NP-Completesa del problema HSTT reduint un problema NP-Complet conegut a HSTT, en aquest cas, el problema de la motxilla[2], d'acord amb la demostració proposada per B. Cooper i J.H. Kingston [3].

Una de les fonts de la duresa del problema ve a l'hora de gestionar els recursos de manera coherent. Així que al confeccionar un horari serà d'interès mantenir el nombre d'incoherències per sota un llindar. Les instànces HSTT acostumen a tenir com a mínim un tipus de recurs que compleix aquestes premises. En cas q no en tinguin, és possible realitzar una transformació binària per arribar a aquesta formulació, ja que tots els recursos poden atendre a un nombre limitat d'assignatures de manera simultània i totes les instàncies disposen d'un nombre limitat de recursos. Per tant, i per simplificar, considerarem només un únic recurs, de disponibilitat limitada. També considerarem que un recurs només pot atendre a una assignatura alhora.

Diem que dues assignatures són incoherents si comparteixen algun espai de temps. És a dir, si es solapen. Com que els recursos són limitats interessa limitar el nombre de solapaments. Utilitzarem una codificació del problema de la motxilla per representar aquesta situació.

El problema de la motxilla consisteix en determinar si un conjunt d'ítems $U = \{u_1, u_2, ..., u_n\}$, cadascun amb un pes associat w_i, es poden col·locar en un conjunt de motxilles $B = \{b_1, b_2, ..., b_m\}$, cadascuna amb una capacitat màxima c_i de manera que cap motxilla sobreexcedeixi la seva capacitat.

Transformem el problema de la motxilla al següent problema HSTT:

$$Times = \{t_1, 1, ..., t_n, m\}$$

$$Events = X \cup Y, X = \{x_1, ..., x_n\} Y = \{y_1, ..., y_m\}$$

De manera que:

- A cada x_i se li han d'assignar tants espais de temps com w_i.
- A cada y_i se li han d'assignar tants espais de temps com c_i i els corresponents a la motxilla que representen. És a dir, y_i tindrà assignats els espais de temps $\{t_i, 1...t_{i,c_i}\}$

El problema doncs, rau en determinar quins espais de temps s'assignen a cada x_i. Suposem que aquest problema formulat com el de la motxilla té solució: $f : U \to B$ on el valor de retorn de f és l'index de la motxilla on s'ha de col·locat l'ítem d'entrada. Llavors, per cada assignatura x_i, escollim w_i espais de temps, q no hagin estat escollits prèviament, del conjunt $S_k = \{t_k, 1...t_{k,c_k}\}$, on $k = f(u_i)$. Es a dir, s'escullen tants espais de temps com el pes de l'ítem que representa de manera que no hi hagin solapaments entre els membres de X. Aquest procés és possible perquè f ens garanteix que com a molt s'escolliran c_k espais de temps de S_k. Al final tenim que tots els events tenen assignats exactament w_i espais de temps i cada X_i se solapa amb un, i només un event de Y. Per tant, tenim que el nombre d'incoherències o solapaments és n.

Ara, suposem que la instància que la instància HSTT que hem descrit genera una solució amb un nombre de solapaments $\leq n$. Sabem que com a mínim el nombre de solapaments ha de ser $\geq n$, ja que cada x_i s'ha de solapar com a mínim una vegada amb algun membre de Y. Per tant el nombre de solapaments de la solució generada per la instància HSTT ha de ser exactament n i cada event x_i es solapa només una vegada amb un sol membre de Y. Podríem reimplementar, doncs, f de manera que a partir de la solució obtinguda per la instància HSTT es limiti a esbrinar per cada event u_i, amb quint event y_i es solapa, de manera que $f(u_i) = j$.

4.3. Format XHSTT

Un dels problemes que presenta HSTT és la complexitat que té representar una instància amb totes les possibles restriccions possibles. Per això s'ha optat utilitzar el format genèric de instanciació de HSTT anomenat XHSTT (de *Xml-format High School TimeTabling*) utilitzat per HSEval[4]. Aquest format és obert i preveu l'addició d'elements i restriccions, així que en aquest treball només es tindràn en compte un subconjunt d'ells.

Aquest format utilitza quatre tipus de fills en les instancies:

- *Times* pels espais de temps.
- *Resources* pels recursos.
- *Events* pels events, com ara assignatures.
- *Constraitns* per les restriccions.

4.3.1 Temps, Recursos i Events

Temps
En aquest tipus de fill es defineixen els multiples espais de temps, i opcionalment els grups de espais de temps. Els Grups de espais de temps (*TimeGroups*) poden ser de tres tipus: *Week*, *Day* i *TimeGroup*. Cada grup consisteix en un nom i un identificador.

Un espai de temps es defineix amb la clau *Time*. Aquesta es pot relacionar amb un grup d'espais de temps utilitzant l'identificador d'aquest. Apart d'això té un nom i un identificador.

A continuació un exemple amb el bloc d'espais de temps.

```
<Times>
    <TimeGroups>
        <Day Id="DayId">
            <Name>DayName</Name>
        </Day>
        <Week Id="WeekId">
            <Name>WeekName</Name>
        </Day>
        <TimeGroup Id="TimeGroupId">
            <Name>TimeGroupName</Name>
        </TimeGroup>
```

```
    </TimeGroups>
    <Time Id="TimeId">
        <Name>Time Name</Name>
        <Day Reference="DayId"/>
        <Week Id="WeekId"/>
        <TimeGroups>
            <TimeGroup Reference="TimeGroupId"/>
        </TimeGroups>
    </Time>
<\Times>
```

Recursos
Cada recurs necessita d'un tipus, com ara aules, professors, classes, etc. Aquests es poden definir amb *ResourceTypes*. A més a més, de la mateixa forma que amb els espais de temps, es poden definir grups de recursos, però a part dels paràmetres mencionats amb els grups de temps, s'ha de incloure el tipus de recursos que inclou.

A continuació un exemple:

```
<Resources>
    <ResourceTypes>
        <ResourceType Id="Room">
            <Name>Room</Name>
        </ResourceType>
    </ResourceTypes>
    <ResourceGroups>
        <ResourceGroup Id="Rooms">
            <Name>Rooms</Name>
            <ResourceType Reference="Room"/>
        </ResourceGroup>
    </ResourceGroups>
    <Resource Id="Room1">
        <Name>Room1</Name>
        <ResourceType Reference="Room"/>
        <ResourceGroups>
            <ResourceGroup Reference="Rooms"/>
        </ResourceGroups>
    </Resource>
</Resources>
```

Events
Al definir un Event cal especificar els tipus de recursos que necessita (es poden assignar recursos concrets), la seva duració i, opcionalment, a quin grup d'events pertany. A més, es pot afegir el rol que té cada recurs en aquest event.

A continuació un exemple:

```
<Events>
    <EventGroups>
        <Course Id="gr_T1-S1">
            <Name>T1-S1</Name>
        </Course>
        <EventGroup Id="gr_AllEvents">
            <Name>All Events</Name>
        </EventGroup>
    </EventGroups>
    <Event Id="T1-S1">
        <Name>T1-S1</Name>
        <Duration>3</Duration>
        <Course Reference="gr_T1-S1"/>
        <Resources>
            <Resource Reference="S1">
                <Role>Class</Role>
                <ResourceType Reference="Class"/>
            </Resource>
            <Resource Reference="T1">
                <Role>Teacher</Role>
                <ResourceType Reference="Teacher"/>
            </Resource>
        </Resources>
        <EventGroups>
            <EventGroup Reference="gr_AllEvents"/>
        </EventGroups>
    </Event>
</Events>
```

4.3.2 Restriccions

Aquí s'enumeraràn els diferents tipus de restriccions definides en el format.[1]

Com a pautes generals, cada restricció tindrà els següents camps:

- *Name*: un nom.
- *Required*: ens diu si el generador té permés (*false*) o no (*true*) violar la restricció . O sigui, si és una *Soft Constraint* o una *Hard Constraint*.
- *Weight*: ens indica el pes de la restricció.
- *CostFunction*: Ens indica la funció que segueix el cost.
- *AppliesTo*: Grups o Elements als quals s'aplica la restricció.

[1] Més informació a `http://www.it.usyd.edu.au/~jeff/cgi-bin/hseval.cgi?op=spec&part=constraints`

Assign Time Constraints

Restricció que imposa que no hi hagi espais de temps sense assignar.

A continuació un exemple:

```
<AssignTimeConstraint Id="AssignTimes">
    <Name>AssignTimes</Name>
    <Required>true</Required>
    <Weight>1</Weight>
    <CostFunction>Linear</CostFunction>
    <AppliesTo>
        <EventGroups>
            <EventGroup Reference="gr_AllEvents"/>
        </EventGroups>
    </AppliesTo>
</AssignTimeConstraint>
```

Split Events Constraints

Restricció que indica com s'han de partir les diferents assignatures, indicant la duració mínima i màxima de cada impartició d'un event o grup d'events.

A continuació un exemple:

```
<SplitEventsConstraint Id="SplitEventsConstraint">
    <Name>Split events to duration 1 and 2</Name>
    <Required>true</Required>
    <Weight>1</Weight>
    <CostFunction>Linear</CostFunction>
    <AppliesTo>
        <EventGroups>
            <EventGroup Reference="gr_AllEvents"/>
        </EventGroups>
    </AppliesTo>
    <MinimumDuration>1</MinimumDuration>
    <MaximumDuration>2</MaximumDuration>
    <MinimumAmount>1</MinimumAmount>
    <MaximumAmount>999</MaximumAmount>
</SplitEventsConstraint>
```

Distribute Split Events Constraints

Restricció que limita el nombre de lliçons de una duració determinada d'un grup d'events. Per exemple, per fer que totes les lliçons de Matemàtiques siguin de duració 2.

A continuació un exemple:

```
<DistributeSplitEventsConstraint Id="DistributeSplit_1">
    <Name>At least 1 double lesson(s)</Name>
    <Required>false</Required>
    <Weight>1</Weight>
    <CostFunction>Linear</CostFunction>
    <AppliesTo>
        <EventGroups>
            <EventGroup Reference="gr_T1-S1"/>
            <EventGroup Reference="gr_T1-S2"/>
            <EventGroup Reference="gr_T1-S3"/>
            <EventGroup Reference="gr_T3-S1"/>
            <EventGroup Reference="gr_T3-S2"/>
            <EventGroup Reference="gr_T3-S3"/>
            <EventGroup Reference="gr_T4-S1"/>
            <EventGroup Reference="gr_T4-S2"/>
            <EventGroup Reference="gr_T4-S3"/>
            <EventGroup Reference="gr_T8-S1"/>
            <EventGroup Reference="gr_T8-S2"/>
            <EventGroup Reference="gr_T8-S3"/>
        </EventGroups>
    </AppliesTo>
    <Duration>2</Duration>
    <Minimum>1</Minimum>
    <Maximum>1</Maximum>
</DistributeSplitEventsConstraint>
```

Prefer Times Constraints

Restricció que indica temps determinats per a certs events. Per exemple per evitar que events de més de una hora a la última hora del dia i acabin a la primera hora del dia següent.

A continuació un exemple:

```
<PreferTimesConstraint Id="PreferredTimes">
    <Name>Times for duration 2</Name>
    <Required>true</Required>
    <Weight>1</Weight>
    <CostFunction>Linear</CostFunction>
    <AppliesTo>
        <EventGroups>
            <EventGroup Reference="gr_AllEvents"/>
        </EventGroups>
    </AppliesTo>
    <TimeGroups>
        <TimeGroup Reference="gr_TimesDurationTwo"/>
    </TimeGroups>
```

```
    <Duration>2</Duration>
</PreferTimesConstraint>
```

Spread Events Constraints

Restricció que indica que els events d'un grup concret san de separar en el temps.

A continuació un exemple:

```
<SpreadEventsConstraint Id="SpreadEvents_2">
    <Name>Spread events max 1 per day</Name>
    <Required>true</Required>
    <Weight>1</Weight>
    <CostFunction>Linear</CostFunction>
    <AppliesTo>
        <EventGroups>
            <EventGroup Reference="gr_T1-S1"/>
            <EventGroup Reference="gr_T1-S2"/>
            <EventGroup Reference="gr_T1-S3"/>
        </EventGroups>
    </AppliesTo>
    <TimeGroups>
        <TimeGroup Reference="gr_Mo">
            <Minimum>0</Minimum>
            <Maximum>1</Maximum>
        </TimeGroup>
        <TimeGroup Reference="gr_Tu">
            <Minimum>0</Minimum>
            <Maximum>1</Maximum>
        </TimeGroup>
    </TimeGroups>
</SpreadEventsConstraint>
```

Avoid Clashes Constraint

Restricció que especifica que cap dels recursos al que s'aplica pot assistir a més d'un event a la vegada. Cal notar que el format permét que un recurs pugui assistir a més dun event a la vegada.

A continuació un exemple:

```
<AvoidClashesConstraint Id="NoResourceClashes">
    <Name>NoResourceClashes</Name>
    <Required>true</Required>
    <Weight>1</Weight>
    <CostFunction>Linear</CostFunction>
    <AppliesTo>
```

```
        <ResourceGroups>
            <ResourceGroup Reference="gr_Teachers"/>
            <ResourceGroup Reference="gr_Classes"/>
        </ResourceGroups>
    </AppliesTo>
</AvoidClashesConstraint>
```

Avoid Unavailable Times Constraints

Restricció que indica que hi ha certes hores durant les quals certs recursos no estan disponibles. Útil per a professors que no treballen cert día o prefereixen no fer-ho en certes hores.

A continuació un exemple:

```
<AvoidUnavailableTimesConstraint Id="AvoidUnavailableTimes_T1">
    <Name>ForbiddenTimesOfT1</Name>
    <Required>true</Required>
    <Weight>1</Weight>
    <CostFunction>Linear</CostFunction>
    <AppliesTo>
        <Resources>
            <Resource Reference="T1"/>
        </Resources>
    </AppliesTo>
    <Times>
        <Time Reference="We_1"/>
        <Time Reference="We_2"/>
        <Time Reference="We_3"/>
        <Time Reference="We_4"/>
        <Time Reference="We_5"/>
    </Times>
</AvoidUnavailableTimesConstraint>
```

Limit Idle Times Constraint

Restricció que límita el número d'espais de temps en qué un recurs o grup de recursos no està ocupat.

A continuació un exemple:

```
<LimitIdleTimesConstraint Id="noIDLETimesT">
    <Name>No IDLE times for teachers</Name>
    <Required>false</Required>
    <Weight>3</Weight>
    <CostFunction>Linear</CostFunction>
    <AppliesTo>
        <ResourceGroups>
```

```xml
            <ResourceGroup Reference="gr_Teachers"/>
        </ResourceGroups>
    </AppliesTo>
    <TimeGroups>
        <TimeGroup Reference="gr_Mo"/>
        <TimeGroup Reference="gr_Tu"/>
        <TimeGroup Reference="gr_We"/>
        <TimeGroup Reference="gr_Th"/>
        <TimeGroup Reference="gr_Fr"/>
    </TimeGroups>
    <Minimum>0</Minimum>
    <Maximum>0</Maximum>
</LimitIdleTimesConstraint>
```

Cluster Busy Times Constraint

Restricció que limita el nombre d'hores en que un recurs pot estar ocupat.

A continuació un exemple:

```xml
<ClusterBusyTimesConstraint Id="MaxNofDaysConstraint_T_days_2">
    <Name>Not more than 2 days with lessons</Name>
    <Required>false</Required>
    <Weight>9</Weight>
    <CostFunction>Linear</CostFunction>
    <AppliesTo>
        <Resources>
            <Resource Reference="T1"/>
            <Resource Reference="T2"/>
            <Resource Reference="T3"/>
            <Resource Reference="T4"/>
            <Resource Reference="T5"/>
            <Resource Reference="T7"/>
            <Resource Reference="T8"/>
        </Resources>
    </AppliesTo>
    <TimeGroups>
        <TimeGroup Reference="gr_Mo"/>
        <TimeGroup Reference="gr_Tu"/>
        <TimeGroup Reference="gr_We"/>
        <TimeGroup Reference="gr_Th"/>
        <TimeGroup Reference="gr_Fr"/>
    </TimeGroups>
    <Minimum>0</Minimum>
    <Maximum>2</Maximum>
</ClusterBusyTimesConstraint>
```

4.4. Estat de l'art

En aquest apartat del treball es repassaran diverses tecnologies i conceptes que ens podrien ajudar a resoldre el problema HSTT. Com s'ha vist anteriorment HSTT és un problema de *scheduling* que pertany a NP-Complet i, per tant, no es coneix un mètode determinista en temps polinòmic que pugui resoldre el problema.

Existeixen diferents aproximacions per problemes del tipus de HSTT, en aquest treball es centrerà en els mètodes deterministes que guaranteixen la optimalitat. Aquests mètodes busquen la millor solució al problema entre totes les combinacions que respecten totes les condicions imposades per la instància del problema que volem resoldre. HSTT entra en la categoria de tècniques basades en programació per restriccions i les tècniques basades en reduccions de altres problemes NP-complets com SAT, SMT, etc.

4.4.1 Problemes de Satisfacció de Restriccions

Els problemes de Satisfacció de Restriccions (CSP de *Constraint Programming Problem*) són una representació de problemes combinatoris. Un CSP està format per un conjunt finit de variables, cada una de les quals te un domini, i un conjunt de restriccions. Cada restricció esta definida sobre un subconjunt de les variables i en restringeix els valors que poden agafar. La idea és trobar una assignació de variables que compleixi totes les restriccions imposades. En alguns problemes, l'objectiu es trobar-les totes, o trobar la millor, si hi ha alguna forma de determinar quines solucions són millors que d'altres utilitzant una formula objectiu.

4.4.2 SAT

SAT probé de SATISFIABILITY, que és la abrebiació de *Boolean Satisfiability Problem.* El problema SAT consisteix en, donada una fòrmula de lògica proposicional (una expressió booleana), determinar una assignació de variables (model) per el qual la fórmula sigui certe, o la determinació de que no existeix tal assignació. Per exemple, per a la fòrmula $A \vee \neg B$ és satisfactible amb $A = Cert$ i $B = Fals$ posat que farien la fòrmula certa; en canvi per la fòrmula $A \vee \neg A$ no hi ha cap assignació que la faci certa, per tant en diriem insatisfactible. SAT consisteix doncs, en determinar si una fòrmula booleana és o no satisfactible.

SAT va ser el primer problema en ser demostrat que era NP-Complet el 1971 per Steven Cool[5], per tant, tal i com s'ha dit anteriorment, tot problema NP es pot reduir a SAT. El fet de que es pot reduir qualsevol problema decidible a SAT en temps polinomíc i la seva simplicitat de formulació el fan un problema d'especial interès en la comunitat científica, generant grans quantitats d'avenços en aquest camp, però, òbviament, tot i així, encara no existeix cap forma de resoldre'l en temps polinomíc ni s'ha demostrat que es pugui (això seria demostrar si P=NP o no![2] per més informació sobre P=NP)

Representació formal, CNF

[2] https://en.wikipedia.org/wiki/P_versus_NP_problem

CNF prové de *Conjunctive Normal Form* que es tradueix com a Forma Normal Conjuntiva. En lògica booleana una fòrmula esta en CNF si és una conjunció de una o més clàusules i aquestes són una disjunció de literals. O sigui, una AND de ORs. Tota fòrmula proposocional pot ser transformada a CNF. Aquesta transformació es basa en les regles d'equivalències lògiques: la doble negació, les lleis De Morgan i la llei de distributivitat.

4.4.3 Extensions de SAT

4.4.3.1 MaxSAT

MaxSAT de *Maximum SATisfiability problem*és una generalització de SAT que consisteix en trobar el màxim nombre de clàusules d'una fòrmula booleana en CNF que es poden satifer. Es pot definir una versió de MaxSAT amb pesos: donada una fòrmula CNF assignem pesos no negatius a cada clàusula i busuqem la assignació de variables que maximitzen el pes sumat de les clàusules satisfetes. Es pot considerar que MaxSAT seria una instància de aquesta versió on tots els pesos son 1.

4.4.3.2 SMT

De *Satisfiability Modulo Theories*, SMT és una generalització de SAT on algunes de les variables proposicionals tenen el paper de predicats amb interpretacions predefinides a d'altres teories. Existeixen diversos tipos de teories: d'igualtat, d'aritmètica lineal, entera, d'aritèmtica lineal mixta, d'arrays, de BitVectors, etc.

Exemple de fòrmula SMT: $p \vee q \vee (x < 5) \vee (y < x)$

Una teoria es defineix com a conjunt de fòrmules lògiques de primer ordre tencades sota conseqüència booleana, o sigui, s'han de poder reduir a un resultat booleà.

4.4.4 Cardinality Encodings

5. Requisits del sistema

En aquest treball es pretén desenvolupar un generador de horaris automàtic amb SAT i/o SMT que serà capaç de rebre una instància XHSTT en un fitxer xml passat per paràmetre, llegir-lo, codificar-ne un model utilitzant la API SMT desenvolupada pel Dr. Jordi Coll, resoldre'l i retornar un resultat, mostrant-lo per pantalla, o retornant un fitxer amb la instància XHSTT amb la nova solució insertada.

Per fer-ho, serà necessari l'ús de un conjunt de llibreries i programari:

- Llibreria que permeti el tractament de fitxers XML, S'ha optat per pugixml[1]
- Posat que la API SMT utilitza Yices 2[2] com a únic solver SMT, necessitarem tenir-ne la seva llibreria instal·lada.
- Posat que la API SMT està desenvolupada per GNU/Linux, serà necessari treballar una màquina que tingui instal·lat una distribució Linux.

Apart de tot el mencionat, també ha calgut decidir un país per el qual el generador d'horaris estarà desenvolupat. Això és degut a qué en funció del país els requisits dels horaris generats varien àmpliament en funció de les característiques pròpies del sistema educatiu de cada país. En aquest treball s'ha decidit optar per les instàncies de Brazil.

[1] https://pugixml.org/
[2] https://yices.csl.sri.com/

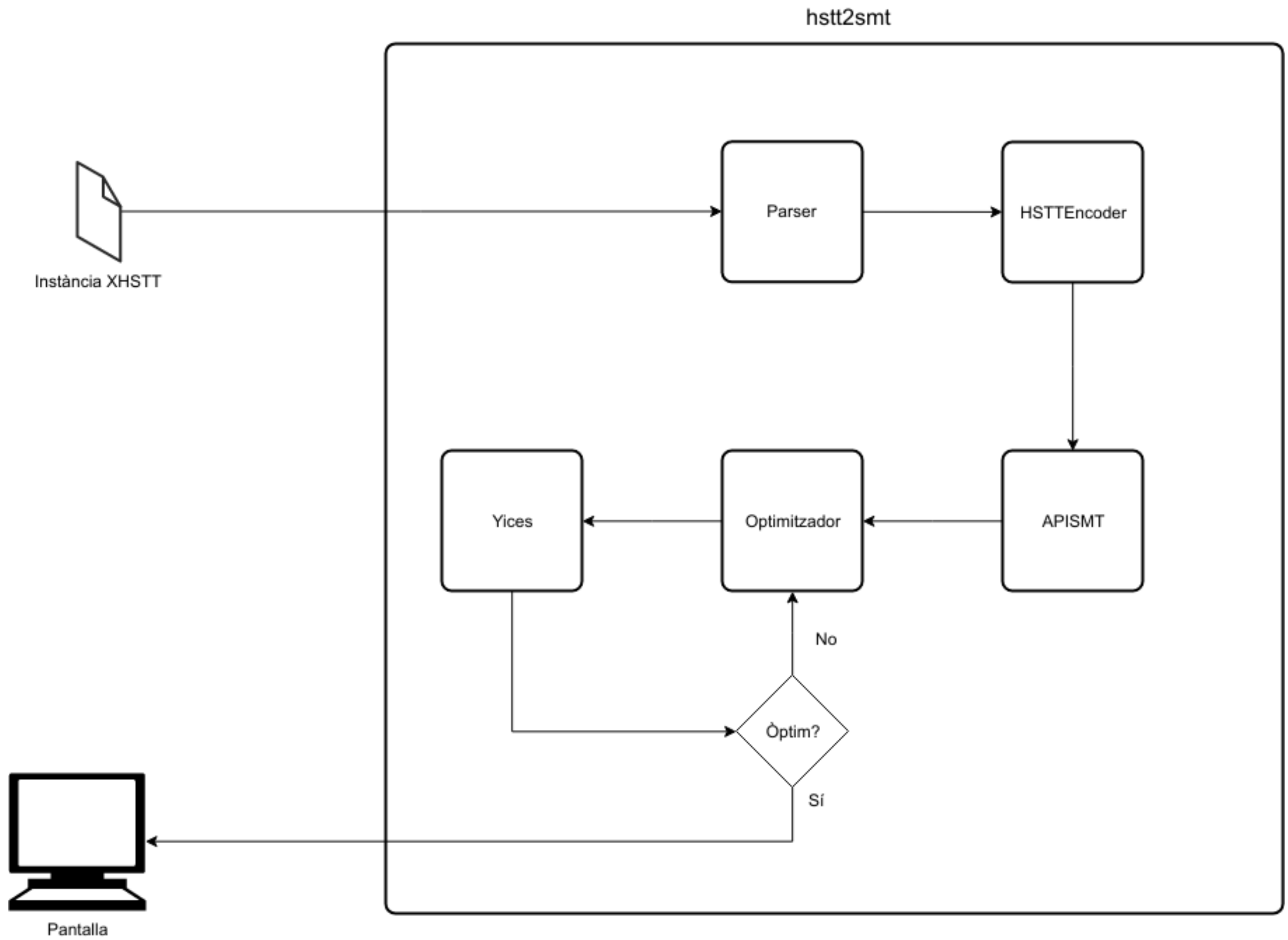

Figura 5.1: Arquitectura del programa

6. Estudis i decisions

6.1. Programari utilitzat

En aquesta secció del treball es veurà tot el programari que s'ha utilitzat per la confecció d'aquest.

6.1.1 Yices 2

Figura 6.1: Logo Yices 2

El Yices 2 és un *SMT Solver Open Source*[1] sota llicència GPL que decideix la satisfactibilitat de fòrmules que contenen símbols de funcions no interpretades amb igualtat, aritmètica real i entera, BitVectors, tipus escalars i tuples. El Yices 2 suporta aritmètica linear i no linear.

El Yices 2 pot processar fitxers d'entrada en notació SMT-LIB[2], es pot usar alternativament el llenguatge propi del Yices 2 i també te una API per C i C++.

En la implementació d'aquest treball, s'usa dins de la API SMT del Dr. Jordi Coll com un dels *solvers* que es poden utilitzar i és l'ùnic que permet SMT d'entre ells. S'ha decidit usar aquest a que és dels millors que hi ha i el grup de recerca hi te experiència prèvia.

S'ha utilitzat la versió 2.6.1.

6.1.2 pugixml

El pugixml és una llibreria C++ lleugera per al processament XML. És extremadament portable amb distribucions. També és opensource[3] sota llicència MIT.

pugixml permet un processament de documents XML molt ràpid, còmode i eficient amb la memòria. Tot i això, al tenir un *parser* DOM, no pot processar fitxers XML que no quèpiguen a la memòria.

[1] Repositori GitHub: `https://github.com/SRI-CSL/yices2`
[2] `http://smtlib.cs.uiowa.edu/`
[3] Repositori GitHub: `https://github.com/zeux/pugixml`

Degut a les caràcterístiques mencionades i a experiència prèvia amb la llibreria, s'ha decidit utilitzar-la per a la programació del *parser* XHSTT.

S'ha utilitzat en la versió 1.9-1.

6.1.3 C++

Figura 6.2: Logo C++

Llenguatge de programació de propòsit general creat per Bjarne Stroustrup com a extensió del llenguatge de programació C. Des de llavors, el llenguatge s'ha expandit molt i el C++ permet programació orientada a objectes, genèrica i funcional, a més de facilitats per manipulació de memòria a baix nivell. Pràcticament sempre és implementat com a llenguatge compilat.

El compilador per a C++ que s'ha utilitzat es el G++ inclòs en el GNU Compiler Collection, que inclou compiladors per a C, C++, Objective-C, Fortran, Ada, Go i D, a més de llibreries per aquests llenguatges.

Figura 6.3: Logo G++

En aquest treball s'ha decidit utilitzar el C++ perquè és amb el que es treballa al grup de recerca i és amb el que està feta la API SMT. El C++ s'ha utilitzat en la revisió C++17 del seu estandard. El GCC utilitzat ha estat en la versió 9.1.0-2.

6.1.4 QtCreator

El QtCreator és un Entorn de Desoenvolupament Integrat (IDE de *Integrated Development Environment*) multiplataforma per a C++, JavaScript i QML que forma part del SDK per el *framework* de desenvolupament d'aplicacions amb GUi Qt. Utilitza

Figura 6.4: Logo QtCreator

S'ha utilitzat la versió 4.9.2-3.

6.1.5 LaTeX

Figura 6.5: Logo LaTeX

Aquesta memòria s'ha confeccionat amb LaTeX, que és un sistema de composició d'alta qualitat que inclou funcions dissenyades per a la creació de documents tècnics i científics amb la intenció de ajudar al creador a centrar-se més en el contingut que en la forma. LaTeXés l'estàndard de facto per a la comunicació i publicació de documents científics.

Per la confecció i edició del document en sí, s'ha utilitzat l'editor Open Source[4] Visual Studio Code. S'ha utilitzat en la versió 1.37.1-2 amb extensions per a facilitar l'edició del document tex.

6.1.6 GNU/Linux

Figura 6.6: Logo Linux

Linux és una familia de sistemes operatius formats pel *kernel* Linux juntament amb les utilitats GNU. El generador que s'ha fet en aquest treball ha estat desenvolupat per a linux, degut a la facilitat

[4] Link GitHub: `https://github.com/Microsoft/vscode`

d'accés a les múltiples llibreries que s'han utilitzat, a que la API SMT ha estat desenvolupada també per a Linux i simplement per prèferencia personal, degut a que el sistema operatiu que utilitzo dia a dia és una distribució Linux.

En concret s'ha treballat amb la distribució Arch Linux en la versió del kernel Linux 5.29.arch1-1 al final del treball (s'han usat versions anteriors que han anat sortint mentre es desenvolupava el treball).

6.2. Maquinari utilitzat

Per efectuar les proves de rendiment s'ha utilitzat un ordinador amb les següents especificacions:

- Processador AMD Ryzen™ 3 1200 a 3.5 GHz amb 4 nuclis físics, 10MB de memòria cau i arquitectura 64 bits.
- 16GB de memòria RAM DDR4 a 3333MT.
- Sistema operatiu Arch Linux 64 bits amb kernel Linux 5.2.9.arch1-1

7. Anàlisis i disseny del sistema

7.1. Anàlisis

7.1.1 Necessitats del sistema

Les necessitats principals del sistema són les següents:

- Necessitem rebre un fitxer de l'usuari.
- Necessitem llegir les dades de un fitxer XHSTT tal i com s'ha explicat anteriorment, per tant requerirem de un *parser* per fer-ho.
- Necessitarem guardar les dades i les restriccions de alguna manera.
- Necessitarem un model lògic i codificar-lo utilitzant la API SMT del Dr. Jordi Coll.
- Necessitarem processar i guardar les dades de manera que ens faciliti el mostrar-les de forma que es pugui entendre.
- Necessitarem comprovar en la mesura del possible que la instància XHSTT sigui correcta.

7.1.2 Anàlisis de processos

L'usuari cridarà el programa, el programa llegirà el fitxer XHSTT que li ha passat l'usuari per paràmetre i s'encarregarà de codificar el model pel yices i cridar-lo per resoldre'l, utilitzant la llibreria per C++ pròpia del yices.

7.2. Disseny

7.2.1 Interfícies d'usuari

El programa funcionarà via consola. Les dades necessàries, com ara el fitxer amb la instància, es passaran per paràmetres, utilitzant el sistema existent en la API SMT del Dr. Jordi Coll.

7.2.2 Model de dades

El model de dades correspondria al següent diagrama:

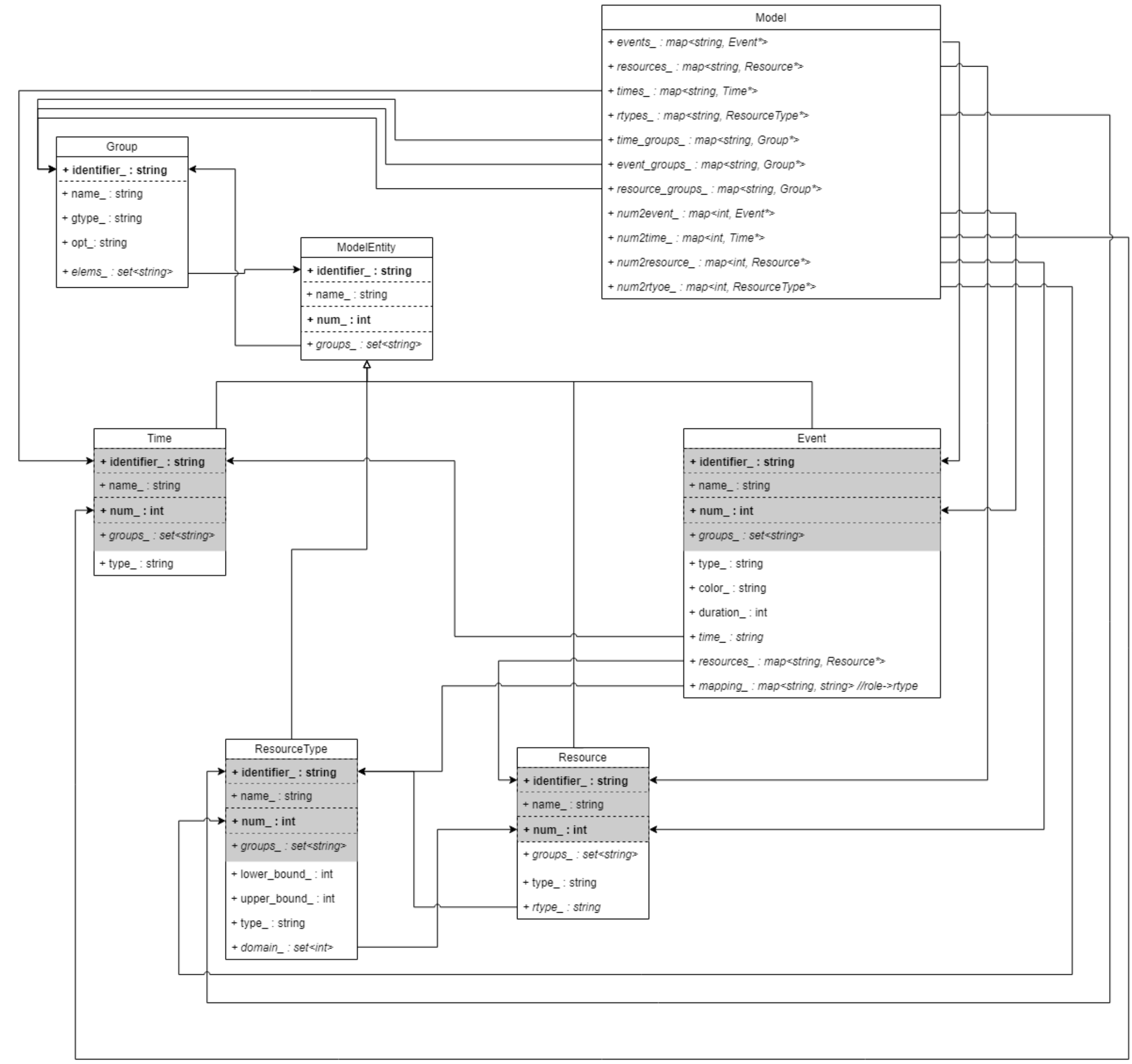

Figura 7.1: Model de Dades

7.2.3 Model d'objectes

El model d'objectes correspondria al següent diagrama:

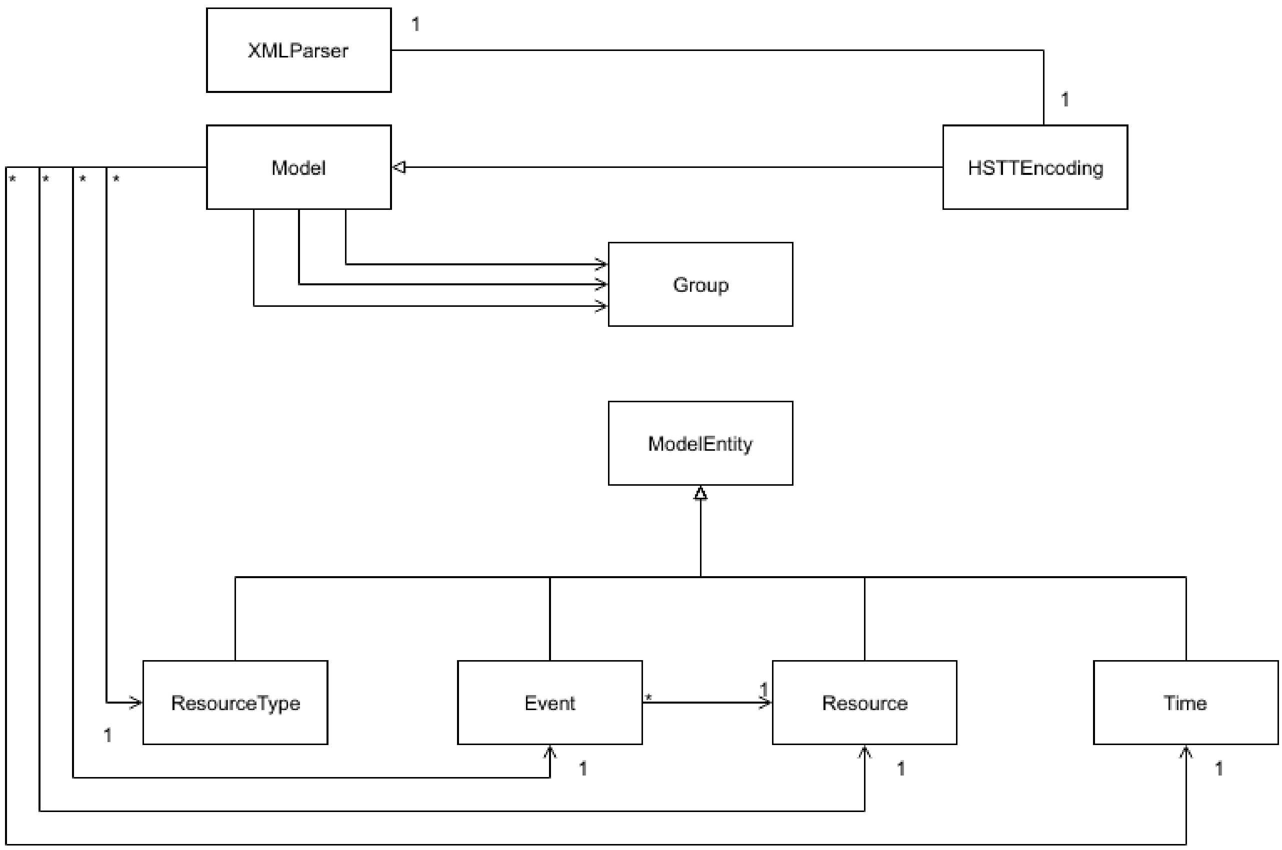

Figura 7.2: Model d'objectes

8. Implementació i proves

9. Implantació i resultats

10. Conclusions

11. Treball futur

12. Manual d'usuari i instal·lació

Bibliografia

[1] Cristòfor Nogueira Gascons. “Generador d’horaris d’instituts”. A: (2015). Disponible a: http://hdl.handle.net/10256/11507.

[2] Wikipedia contributors. *Bin packing problem — Wikipedia, The Free Encyclopedia*. 2019. URL: https://en.wikipedia.org/w/index.php?title=Bin_packing_problem&oldid=912207075.

[3] Tim B Cooper i Jeffrey H Kingston. “The Complexity of Timetable Construction Problems Technical Report Number 495”. A: (febr. de 1995).

[4] Jeffrey H. Kingston. *High School Timetable Data Format Specification*. 2012. URL: http://www.it.usyd.edu.au/~jeff/cgi-bin/hseval.cgi?op=spec.

[5] Stephen A Cook. “The complexity of theorem-proving procedures”. A: *Proceedings of the third annual ACM symposium on Theory of computing*. ACM. 1971.

[6] Universitat de Twente. *International High School Timetabling competition*. 2018. URL: http://www.utwente.nl/ctit/hstt/.

[7] Aishwarya Agarwal. *Theory of computation — Decidable and undecidable problems — GeeksforGeeks*. URL: https://www.geeksforgeeks.org/theory-computation-decidable-undecidable-problems/.

[8] GeeksforGeeks. *Theory of computation — Decidable and undecidable problems — GeeksforGeeks*. URL: https://www.geeksforgeeks.org/np-completeness-set-1/.

[9] Arseny Kapoulkine. *pugixml*. URL: https://pugixml.org/.

[10] SRI International. *The Yices SMT Solver*. URL: https://yices.csl.sri.com/.

[11] Standard C++ Foundation. *Standard C++*. URL: https://isocpp.org/.

[12] Wikipedia contributors. *C++ — Wikipedia, The Free Encyclopedia*. [Online; accessed 29-August-2019]. 2019. URL: https://en.wikipedia.org/w/index.php?title=C%2B%2B&oldid=912907129.

[13] The Qt Company. *Qt APIs & Libraries, Tools and IDE*. URL: https://www.qt.io/qt-features-libraries-apis-tools-and-ide/#ide.

[14] The LaTeX project. *LaTeX– A document preparation system*. URL: https://www.latex-project.org/.

[15] *Visual Studio Code*. URL: https://code.visualstudio.com/.

[16] The Linux Foundation. *Linux*. URL: https://www.linuxfoundation.org/projects/linux/.

[17] ArchWiki. *Arch Linux*. URL: https://wiki.archlinux.org/index.php/Arch_Linux.

[18] Zhe Liu. *Algorithms for Constraint Satisfaction Problems (CSPs)*. Universitat de Waterloo. Disponible a: http://www.cs.toronto.edu/~fbacchus/Papers/liu.pdf. 1998.

[19] Mauricio Toro et al. "GELISP : A Framework to represent musical constraint satisfaction problems and search strategies". A: 86 (abr. de 2016). Disponible a: http://www.jatit.org/volumes/Vol86No2/17Vol86No2.pdf, pàg. 327 - 331.

[20] Wikipedia contributors. *Boolean satisfiability problem — Wikipedia, The Free Encyclopedia.* 2019. URL: https://en.wikipedia.org/w/index.php?title=Boolean_satisfiability_problem&oldid=911344299.

[21] Wikipedia contributors. *Maximum satisfiability problem — Wikipedia, The Free Encyclopedia.* 2019. URL: https://en.wikipedia.org/w/index.php?title=Maximum_satisfiability_problem&oldid=900651688.

[22] Clark Barrett i Cesare Tinelli. "Satisfiability modulo theories". A: *Handbook of Model Checking.* Springer, 2018, pàg. 305 - 343.

[23] Leonardo De Moura i Nikolaj Bjørner. "Satisfiability modulo theories: introduction and applications". A: *Communications of the ACM* 54.9 (2011), pàg. 69 - 77.

[24] Wikipedia contributors. *Satisfiability modulo theories — Wikipedia, The Free Encyclopedia.* 2019. URL: https://en.wikipedia.org/w/index.php?title=Satisfiability_modulo_theories&oldid=909772056.