KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS
INFORMACIJOS SISTEMŲ KATEDRA

Aleksandr Murašov

Daugiamačių duomenų analizės galimybių išplėtimas taikant šablonus

Magistro darbas

Darbo vadovas**:** prof. L. Nemuraitė

Kaunas, 2008

KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS
INFORMACIJOS SISTEMŲ KATEDRA

Aleksandr Murašov

Daugiamačių duomenų analizės galimybių išplėtimas taikant šablonus

Magistro darbas

Recenzentas
dr. A. Janavičiūtė
2008-05-26

Vadovas
Prof. L. Nemuraitė
2008-05-26

Atliko
IFM-2/1 gr. stud.
Aleksandr Murašov
2008-05-26

Kaunas, 2008

Turinys

Summary

This work is related to Business Intelligence technology, data warehouses, OLAP technology and query languages. Using modern data analysis functions we will show an approach to increase the efficiency of data analysis by using multidimensional data query and analysis languages like MDX and DMX. These languages are poorly known and have great potential. For example we can enhance data warehouse queries with Data Mining functions, which can give us very quick and accurate results. We will also discuss possibilities to use DMX patterns for data prediction. We will use Data Mining prediction functions to solve the lack o data in data warehouses problem. It is very unusual usage of Data Mining functions, because such functions are usually used for data analysis, but not for data renewal.

1. Įvadas

Dėl pastoviai augančio informacijos sistemų sudėtingumo ir duomenų kiekio augimo organizacijos susiduria su problema, kaip efektyviai saugoti ir analizuoti istorinius duomenis. Paprastos reliacinės duomenų bazės jau negali patenkinti reikalavimų duomenų analizei, nes jų struktūra dažniausiai yra sudėtinga, dėl ko gali labai išaugti duomenų išrinkimo laikas. Greitas rezultatų pateikimas duomenų analizėje gali būti labai svarbus, nes tai leidžia operatyviai gauti informaciją apie organizacijos veiklą ir priimti svarbius ir teisingus sprendimus. Todėl šiuo metu labai sparčiai vystosi verslo intelekto technologija (*angl. Business Intelligence*), kuri suteikia daug metodų ir sprendimų greitam ir efektyviam veiklos duomenų saugojimui, integravimui, išrinkimui, analizei ir pateikimui. Verslo intelekto sistemos integruojasi su informacinėmis sistemomis ir jau tampa neatsiejama didelių informacinių sistemų dalimi. Paskutiniu metu vis daugiau organizacijų diegia arba planuoja diegti verslo intelekto sistemas.

Kadangi pagrindinė verslo intelekto užduotis yra užtikrinti efektyvią duomenų analizę, tai duomenų saugyklas galima pavadinti verslo intelekto pamatais. Duomenų saugyklą yra tokia duomenų bazė, kurioje saugomi specialiai pagal duomenų analizės reikalavimus transformuoti duomenis. Tokių duomenų struktūra orientuota į greitą duomenų išrinkimą ir analizei būtiną detalumo lygį.

Per paskutinį dešimtmetį sparčiai populiarėjo OLAP (*angl. On Line Analitical Processing*) duomenų analizės technologija, kuri priklauso veiklos intelekto technologijų aibei. Esminės šitos technologijos savybės yra tokios: apdorojami labai dideli duomenų kiekiai, labai greitas duomenų išrinkimas, didelės duomenų agregacijos, duomenys turi daugiamatį pavidalą. Dažniausiai OLAP technologija taikoma pardavimų analizėje, biudžetų analizėje ir prognozėje, finansinės informacijos analizėje, verslo procesų analizėje ir t.t.

Dar naujesnė ir sparčiai besivystanti veiklos intelekto technologijų aibei priklausanti technologija yra duomenų gavyba (*angl. Data Mining*). Tai yra technologija turinti daug duomenų analizei skirtų galingų funkcijų. Duomenų gavybos pagrindinis uždavinys yra išgauti iš duomenų informaciją, kurią sunku aptikti naudojant įprastus analizės metodus. Naudojant šią duomenų analizės technologiją galima spręsti tokius uždavinius kaip klasifikavimas, grupavimas, asociacija, nuokrypių analizė ir prognozavimas.

Sudėjus kartu duomenų saugyklas, OLAP technologiją ir duomenų gavybą, galima pasiekti labai gerų rezultatų. Efektyvus duomenų saugojimas, greitas rezultatų pateikimas ir galingos duomenų analizės funkcijos gali padėti priimti greitus, teisingus ir pagristus veiklos sprendimus.

Šiuo metu rinkoje yra pakankamai daug verslo intelekto įrankių. Pasaulyje labai gerai žinomi Microsoft, Oracle, IBM, SAP, SAS, Teradata, Information Builders ir kitų veiklos intelekto sprendimų kūrėjai. Kadangi veiklos intelekto technologija yra pakankamai jauna, tai jai nėra priimta daug standartų. Dėl to skirtingų veiklos intelekto įrankių kūrėjų produktų lygis labai skiriasi, vieni yra stipresni ir siūlo daugiau galimybių, kiti silpnesni, turi mažiau funkcijų ir net pačių įrankių realizavimo kokybė gali būti prasta. Todėl geriausias būdas stebėti šios srities vystymąsi yra rinkos analizė.

Darbo planas susideda iš penkių pagrindinių punktų: tikslų ir uždavinių apibrėžimas, technologijos ir jos galimybių analizė, sprendimo iškeltai problemai suformulavimas, suformuluoto sprendimo eksperimentinis realizavimas, tyrimo išvados .

Darbo tikslas yra ištirti veiklos intelekto technologijos galimybes, naudojamus metodus, sprendimus ir pritaikyti juos savo probleminei sričiai.

Šiame darbe iškelta problema yra daugiamačių duomenų analizės galimybių išplėtimas trūkstamų duomenų nustatymui, išnaudojant šiuolaikinių technologijų galimybes. Tiksliau tai galima apibrėžti kaip veiklos intelekto funkcijų taikymas duomenų saugykloje trūkstamų duomenų nustatymui ir jų sukūrimui. Šiai problemai išspręsti reikia naudoti skirtingas duomenų išrinkimo kalbas, nes reikia manipuliuoti tiek plokščiais, tiek daugiamačiais duomenimis. Duomenų saugyklų užpildymui naudojami ETL (*angl. Extract, Transform, Load*) procesai, transformuojant operacinės duomenų bazės duomenis ir perkeliant juos į saugyklą. Bet jeigu duomenis operacinėje duomenų bazėje atsiranda tik su tam tikru periodiškumu, tada duomenų saugykloje atsiranda neužpildytos vietos t.y. duomenų trūkumas. Trūkstamų duomenų sukūrimo problemą galima išspręsti panaudojant duomenų gavybos prognozavimo funkcijas, kurios naudoja statistinius prognozavimo metodus. Tai yra neįprastas prognozavimo funkcijos panaudojimas, nes tokios funkcijos dažniausiai naudojamos pardavimų ar panašių rodyklių prognozavimui, o ne trūkstamų duomenų sukūrimui. Reiškia problemos sprendimas galėtų būti prognozavimo funkcijos naudojimas vietoj

ETL proceso, o duomenų šaltinu būtų ne operacinė duomenų bazė, o duomenų saugykla.

Toks sprendimas yra geras tuo, kad gali ženkliai paspartinti problemos sprendimą, nes nereikia realizuoti trūkstamų duomenų sukūrimo funkcijų. Pakanka suprojektuoti vieną universalų prognozavimo funkcijos šabloną, kuris galėtų grąžinti reikiamo detalumo rezultatus ir užtikrinti efektyvų bei korektišką šio šablono naudojimą.

Tyrimams atlikti buvo pasirinktas Microsoft korporacijos produktas Microsoft SQL Server 2005. Pagrindinis tyrimuose naudojamas produkto modulis yra Microsoft Analysis Services 2005. Šis modulis yra veiklos intelekto įrankis, turintis praktiškai visas rinkos naujoves. Jame pilnai realizuotos OLAP ir duomenų gavybos technologijos, ko nėra daugumoje kitų veiklos intelekto įrankių. Dar vienas svarbus faktorius yra tas, kad Microsoft SQL Server 2005 paketas turi visus pilnos veiklos intelekto sistemos realizavimui reikalingus modulius: reliacinių bazių modulis, veiklos intelekto modulis, duomenų integravimo modulis ir duomenų atvaizdavimo (ataskaitų) modulis. Tai yra labai patogu, nes nereikia naudoti skirtingų kūrėjų produktus, kuriuos naudojant gali iškilti tarpusavio integracijos problemų.

Šia tema buvo skaitytas pranešimas tarpuniversitetinėje magistrantų ir doktorantų konferencijoje Informacinės Technologijos 2008.

2. Veiklos intelekto priemonių analizė

2.1.Tyrimo sritis, objektas ir problema

Tyrimo sritis yra veiklos intelekto sistemos, duomenų saugyklos, bei duomenų išrinkimas iš jų. Duomenų saugyklose laikomi labai dideli duomenų kiekiai, t.y. informaciją apie organizacijos veiklą per tam tikrą laikotarpį. Šiuos duomenis naudoja sprendimų priėmimo sistemos, kurios suteikia galimybę analizuoti duomenis, gauti apibendrintus rezultatus apie organizacijos veiklos rezultatus. Remiantis duomenų analizės rezultatais galima daryti tam tikrus verslo sprendimus, priimti tam tikras veiklos strategijas, gaires.

Dirbant su dideliais duomenų kiekiais visada iškyla viena problema – duomenų gavimo laikas. Kuo daugiau duomenų, tuo didesnis jį išrinkimo laikas. Todėl duomenų išrinkimo būdas, arba technologija, yra labai svarbus aspektas. Viena šitos problemos dalis yra duomenų išrinkimo užklausų kalbos. Pavyzdžiui iš duomenų saugyklos duomenis galima išrinkti naudojant ir SQL ir MDX užklausų kalbas. SQL yra universali užklausų kalba, kuri nėra labai patogi ir greita dirbant su daugiamatėmis duomenų struktūromis. MDX užklausų kalba, savo ruožtu, buvo sukurta specialiai darbui su daugiamatėmis struktūromis, todėl yra patogesnė ir greitesnė, bet sudėtingesnė. Būtent dėl to ir reikia atlikti šitos kalbos analizę, ištirti jos ypatumus, galimybes, ribas, bei taikymo aspektus. Taip pat reikia išsiaiškinti kuo MDX skiriasi nuo SQL užklausų kalbos.

2.2.Tyrimo tikslas ir uždaviniai

Bet kurio tyrimo vienas pagrindinių dalių yra iškelti tikslus ir uždavinius, nes tyrimas privalo turėti vienareikšmišką kryptį, ties kurią reikia dirbti. Taigi tyrimo tikslas yra paspartinti duomenų analizei ir veiklos intelektui skirtų sprendimų kūrimą, sukuriant šablonus. Tyrimo metu iškeliami tokie uždaviniai

1. Atlikti duomenų analizės ir veiklos intelekto kalbų SQL, MDX ir DMX analizę, nustatyti jų trūkumus ir privalumus bei taikymo galimybes.
2. Ištirti šių kalbos plėtimo galimybes.
3. Sukurti užklausų šablonus trūkstamų veiklos duomenų nustatymui.
4. Pritaikyti šabloną pavyzdinės saugyklos duomenims nustatyti.
5. Įvertinti sukurto sprendimo efektyvumą..

2.3.Tyrimo planas

Tyrime laikysimės tokio plano:

1. Analizės metodų parinkimas.
2. Literatūros studijavimas.
3. Modeliavimas ir eksperimentų atlikimas.
4. Eksperimentų rezultatų nagrinėjimas.
5. Išvadų sudarymas.

2.4.Analizės tikslas

Analizės tikslas yra įsigilinti į tyrimo sritį, objektą ir iškeltas problemas. Tik iki galo supratus srities specifiką, galima spręsti iškeltas problemas. Taigi analizės tikslai yra tokie:

1. Įsigilinti į tyrimo sritį.
2. Išanalizuoti klausymus, susijusius su tyrimo objekto problemomis.
3. Rasti sprendimus iškeltoms problemoms ir uždaviniams.

Tyrimo objekto analizę atliksime, studijuodami literatūrą bei straipsnius. Šis metodas yra tinkamas todėl, kad literatūroje ir straipsniuose galima rasti daug informacijos apie objekto savybes, galimybes, trūkumus. Išstudijavus literatūrą ir radus informacijos apie tyrime iškeltas problemas, galima daryti kažkokius sprendimus. Mūsų atveju, kai tyrimo objektas yra užklausų kalba, tai yra tinkamas metodas, nes apie tokį objektą informacijos galima gauti tik iš literatūros ir straipsnių.

Be to kai kuriais atvejais naudinga atlikti realų eksperimentą, nes tik taip galima įsitikinti, kad gauta informacija buvo teisinga ir tiksli. Užklausų kalbų palyginimas yra tinkamas analizės metodas, nes taip galima nesunkiai išsiaiškinti kalbų specifikas.

2.5.Duomenų saugyklos

Didelės informacinės sistemos dažniausiai turi papildomus įrankius, su kuriais galima atlikti sudėtingą kompleksinę daugiamatę duomenų analizę. Tokia analizė galiausiai turi padėti sprendimų priėmimui. Dažniausiai tokios sistemos taip ir vadinamos – Sprendimo Priėmimo Sistemos [2].

Neįmanoma priimti bet kuri administruojantį sprendimą neturint tam būtinos informacijos, dažniausiai kiekybinės. Tam būtinai reikia sukurti duomenų saugyklas (*angl. Data Warehouses*). Tai yra duomenų surinkimo, filtravimo, apdorojimo

procesai, kurie reikalingi tam, kad vartotojui būtų pateikiama statistinėms analizėms ir analitinėms ataskaitoms tinkama informacija.

Ralfas Kimbalas (Ralph Kimball) suformulavo pagrindinius reikalavimus duomenų saugykloms [4]:

- Didelis duomenų iš saugyklos gavimo greitis.
- Vidinis duomenų nuoseklumas.
- Galimybė gauti ir lyginti taip vadinamus duomenų griežinėlius (*angl. slice and dice*).
- Turi būti įrankiai, skirti saugykloje esančių duomenų peržiūrai.
- Saugomų duomenų pilnumas ir tapatumas.
- Turi būti geras saugyklos papildymo duomenimis procesas.

Patenkinti visus šiuos reikalavimus viename produkte dažniausiai nepavyksta, todėl duomenų saugyklos realizavimui dažniausiai naudojami keli produktai. Vieni produktai yra duomenų saugojimo priemonės, kiti produktai padeda išgauti ir peržiūrėti duomenis, treti – duomenų įkėlimo produktai.

Tipinė duomenų saugykla skiriasi nuo reliacinės duomenų bazės. Pirmiausiai, paprastos duomenų bazės skirtos tam, kad padėti vartotojams atlikti kasdieninį darbą, o duomenų saugyklos skirtos sprendimų priėmimui. Pavyzdžiui, prekių pardavimai atliekami naudojant transakcijų apdorojimui skirtą duomenų bazę, o pardavimų analizei atlikti naudojama duomenų saugykla. Dirbant vartotojams paprastos duomenų bazės duomenis pastoviai keičiasi, o duomenų saugyklos duomenis yra palyginus stabili: duomenis dažniausiai atnaujinami pagal tvarkaraštį (kas valandą, kas savaitę, kas mėnesį), priklausomai nuo poreikio. Idealiu atveju duomenų saugyklos papildymo procesas yra paprastas tam tikro laiko tarpo informacijos įkėlimas nekeičiant saugykloje esančių duomenų. Paprastos duomenų bazės yra duomenų šaltinis. Šitie duomenis ir patenka į duomenų saugyklas. Be to, duomenų saugyklos gali būti papildomos iš išorinių šaltinių (pavyzdžiui iš statistinių ataskaitų) [6].

Tipinė duomenų saugyklų struktūra

Pagrindinė OLAP idėja yra ta, kad reikia sudaryti daugiamačius kubus, kurie butų prieinami vartotojų užklausoms. Bet OLAP kubu pradiniai duomenis dažniausiai saugomi reliaciniuose duomenų bazėse. Kartais tai būna specializuotos reliacinės duomenų bazės, kurios vadinamos duomenų saugyklomis (*angl. Data Warehouse*).

Duomenų saugyklos reikalingos tik informacijos apdorojimui ir analizei, todėl jos projektuojamos taip, kad užklausų atlikimo laikas būtų minimalus [3].

Tipinė duomenų saugyklos struktūra ženkliai skiriasi nuo paprastos reliacinės duomenų bazės struktūros. Ši struktūra nenormalizuota (kas leidžia sumažinti užklausų atlikimo laiką), todėl gali pasitaikyti duomenų pertekliškumas.

Duomenų saugyklos struktūra susideda iš faktų lentelės ir dimensijos (mato) lentelių.

Faktų lentelė

Faktų lentelė yra pagrindinė duomenų saugykloje. Joje saugoma informacija apie objektus arba įvykius, kurie bus analizuojami. Dažniausiai naudojami faktų tipai yra tokie:

- Faktai susiję su transakcijomis. Jie remiasi atskirais įvykiais (pavyzdžiui telefono skambutis arba pinigų nuėmimas iš sąskaitos pasinaudojus bankomatu).
- Faktai susiję su momentiniais įvykiais. Remiasi objekto būsena tam tikrais laiko momentais. Tipinis tokių faktų pavyzdys yra dienos pardavimų apimties faktas.
- Faktai susiję su dokumento elementais. Remiasi vienokiu ar kitokiu dokumentu (pavyzdžiui sąskaita už prekes arba paslaugas), ir turi išsamią informaciją apie šio dokumento elementus (pavyzdžiui kiekis, kaina, nuolaidos procentai).
- Faktai susiję su įvykiais arba objekto būsena. Parodo įvykio atsiradimą be papildomos informacijos apie jį (pavyzdžiui pardavimo faktas be jokios papildomos informacijos).

Faktų lentelė turi turėti sudėtinį raktą, kuris apima pirminius dimensijos lentelių raktus. Dažniausiai tai būna sveikųjų skaičių arba datos ir laiko reikšmės, juk faktų lentelė gali turėti šimtus tūkstančių arba milijonus įrašų, ir pasikartojančią informaciją geriausiai laikyti mažesnėse faktų lentelėse. O raktiniai ir neraktiniai laukai turi atitikti būsimas OLAP kubo dimensijas. Be to, faktų lentelė turi turėti kelis skaičių laukus, pagal kurios ateityje galima būtų gauti agreguotus duomenis. Faktų lentelėje nėra jokių duomenų kaip reikia grupuoti įrašus skaičiuojant agreguotus duomenis. Pavyzdžiui, joje yra prekių ir klientų identifikatoriai, bet nėra informacijos

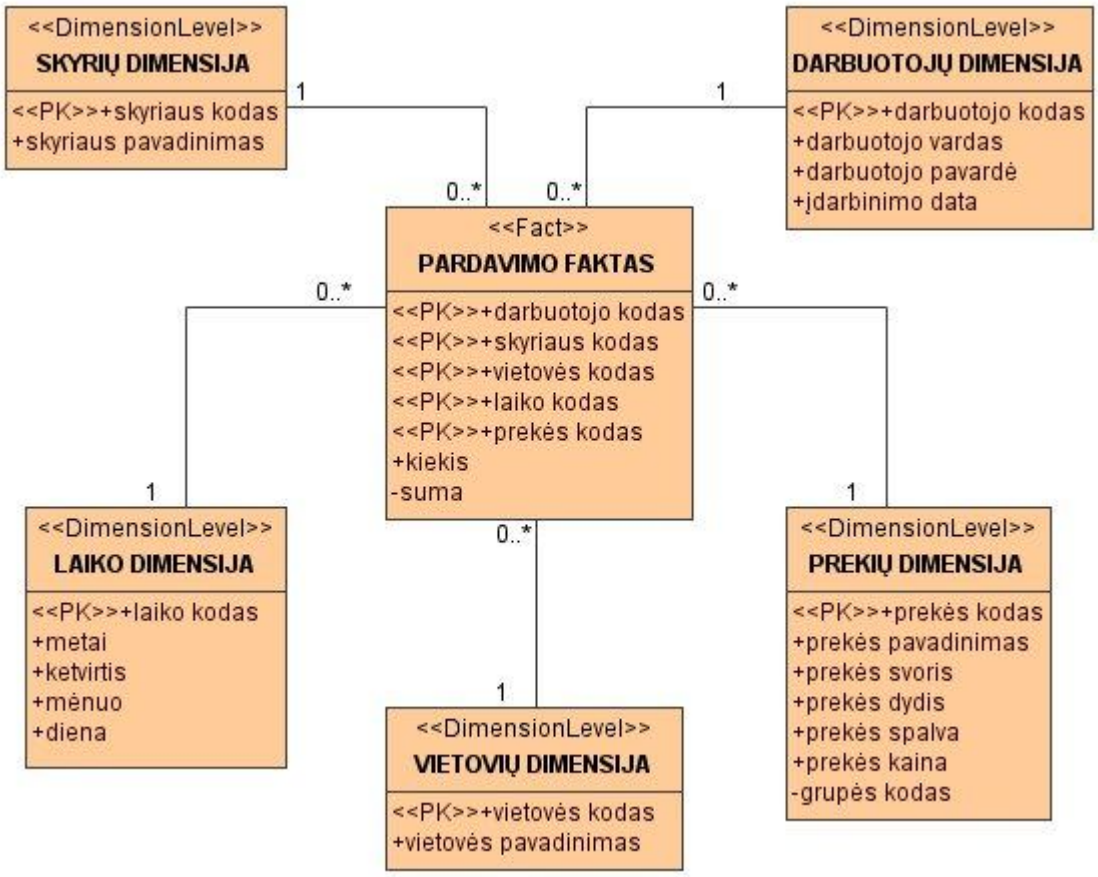
kokiai grupei priklauso prekė arba kur gyvena klientas. Tokie duomenys laikomi dimensijos lentelėse ir naudojami sudarant hierarchijas OLAP kubo dimensijose.

Dimensijų lentelės

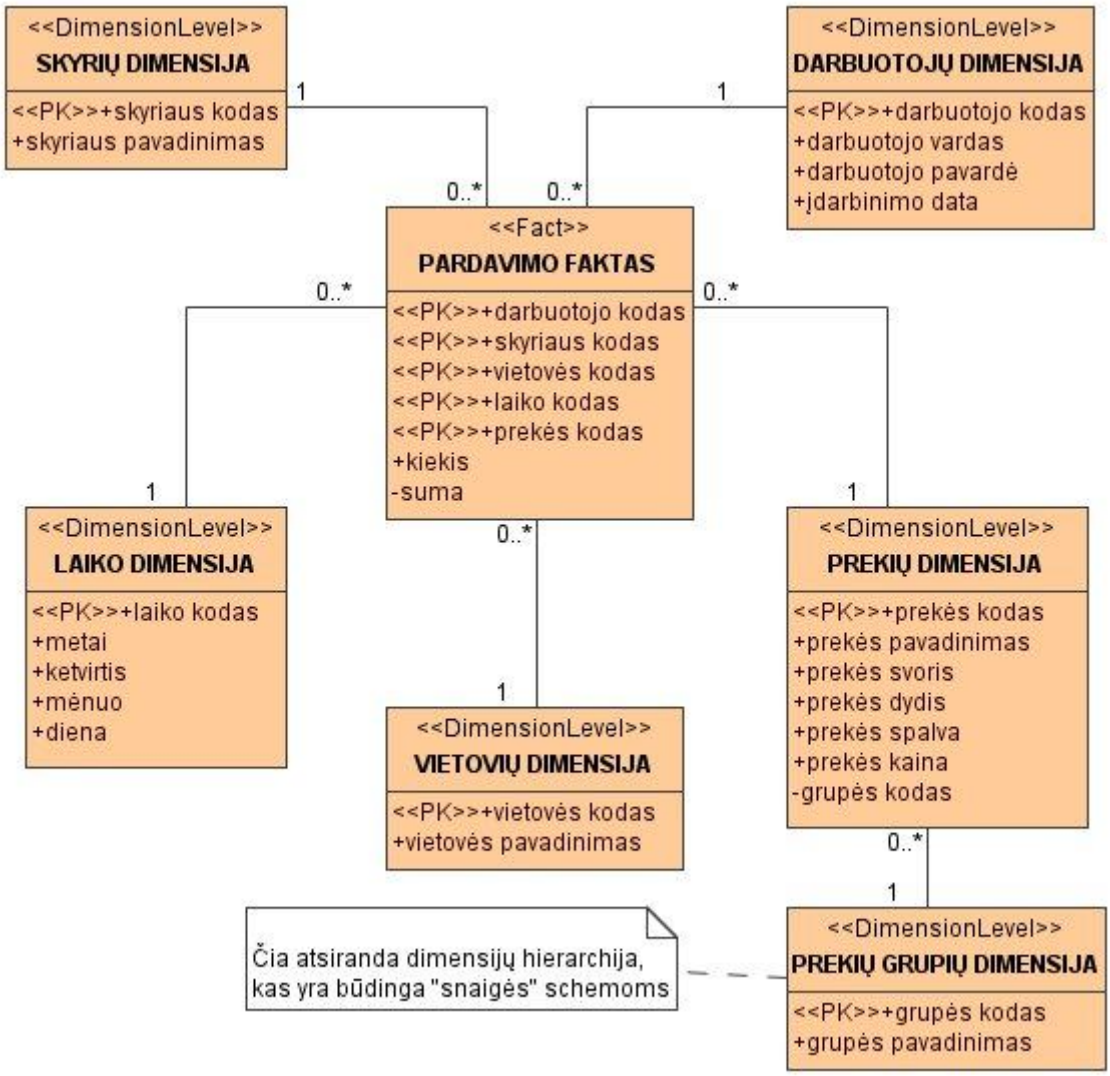
Dimensijos lentelėse laikomi nesikečiantys arba retai besikeičiantys duomenys. Dažniausiai šitie duomenis turi po vieną įrašą kiekvienam šios dimensijos žemesnio hierarchijos lygio nariui. Dimensijos lentelės taip pat turi bent po vieną aprašomąjį lauką (su dimensijos nario pavadinimu) ir skaitinį raktinį lauką vienareikšmiškai dimensijos nario identifikacijai. Jeigu nauja dimensija užima kažkokią vietą hierarchijoje, tai ši lentelė gali turėti laukus, parodančius tėvą šioje hierarchijoje. Kartais dimensijos lentelė gali turėti laukus, parodančius protėvius hierarchijoje (tai būdinga subalansuotoms hierarchijoms), o taip pat papildomus dimensijos narių atributus, kurie buvo laikomi pirminėje darbinėje duomenų bazėje (pavyzdžiui klientų adresai ir telefonų numeriai). Kiekviena dimensijos lentelė su faktų lentele turi turėti kardinalią „vienas su daug“. Pažymėtina, kad dimensijos lentelių augimo greitis yra labai nežymus palygimus su faktu lentele. Pavyzdžiui naujas įrašas į dimensijos lentelėje, charakterizuojančioje prekes, atsiranda tik tada, kai atsiranda visiškai nauja prekė. Viena kubo dimensija gali susidėti iš vienos lentelės (netgi tada, kai gali egzistuoti keli hierarchijos lygiai), arba iš kelių, surištų ir atitinkančių tam tikrą hierarchijos lygį, lentelių.

Saugyklų schemų tipai

Jeigu visos dimensijos susideda ne daugiau nei iš vienos lentelės, tai duomenų saugyklos schema vadinama žvaigždė (*angl. star schema*, pav. 1). Jeigu bent viena dimensija susideda iš kelių surištų lentelių, tai tokia duomenų saugyklos schema vadinama snaigė (*angl. snowflake schema*, pav. 2). Papildomos dimensijos lentelės tokioje schemoje, kurios dažniausiai atitinka viršutinius hierarchijos lygius ir su pagrindinę dimensijų lentele, atitinkančia žemesnį hierarchijos lygį, turi kardinalumą vienas su daug, kartais vadinamos konsulinėmis lentelėmis (*angl. outrigger table*). Pažymėtina, kad dažniausiai, siekiant sumažinti užklausų atlikimo laiką, geriau naudoti žvaigždės tipo duomenų saugyklas [6].



pav. 1. Žvaigždės schema



pav. 2. Snaigės schema

Bet ne visos duomenų saugyklos projektuojamos pagal šitas dvi schemas. Taip, faktų lentelėje, vietoj raktinio lauko, susiejančio ją su dimensijos lentele, gali būti raktinis laukas, turintis konkrečią reikšmę. Tokiu atveju dimensijos lentelės gali ir nebūti.

Nesubalansuotos hierarchijos atveju į snaigės schemą taip pat reikėtų įvesti pakeitimus. Tokiu atveju dimensijos lentelėje yra ryšys, analogiškas ryšiui operatyvioje duomenų bazėje.

Dar vienas taisyklių pažeidimų pavyzdys gali būti toks: vienoje dimensijoje gali būti kelios hierarchijos. Tipiniai pavyzdžiai yra tokie: hierarchijos kalendoriniams ir finansiniams metams (su sąlyga, kad finansiniai metai prasideda ne nuo sausio pirmos dienos), arba su skirtingais dimensijos nariu grupavimais (pavyzdžiui, prekes galima grupuoti pagal kategorijas, arba pagal gamintojus, arba tiekėjus). Tokiu atveju

dimensijos lentelė turi laukus visoms įmanomos hierarchijoms, su vienodais žemesnio lygio nariais, bet su skirtingais aukštesnių lygių nariais.

Kaip jau buvo minėta, dimensijos lentelė gali turėti laukus, nesusijusius su hierarchijomis, o turinčius dimensijos narių atributus. Kartais tokie atributai gali būti panaudoti atliekant analizę.

Reikia pažymėti, kad reliacinių duomenų saugyklų kūrimui dažnai naudojamos specializuotos DBVS, kuriose duomenų saugojimas yra optimizuotas užklausų atlikimo atžvilgiu. Tokio produkto pavyzdys galėtų būti Sybase Adaptive Server IQ, kuriame realizuotas netradicinis duomenų saugojimo būdas (ne pagal eilutes, o pagal stulpelius). Bet duomenų saugyklą galimą kurti naudojant ir paprastus reliacinius DBVS.

2.6.OLAP

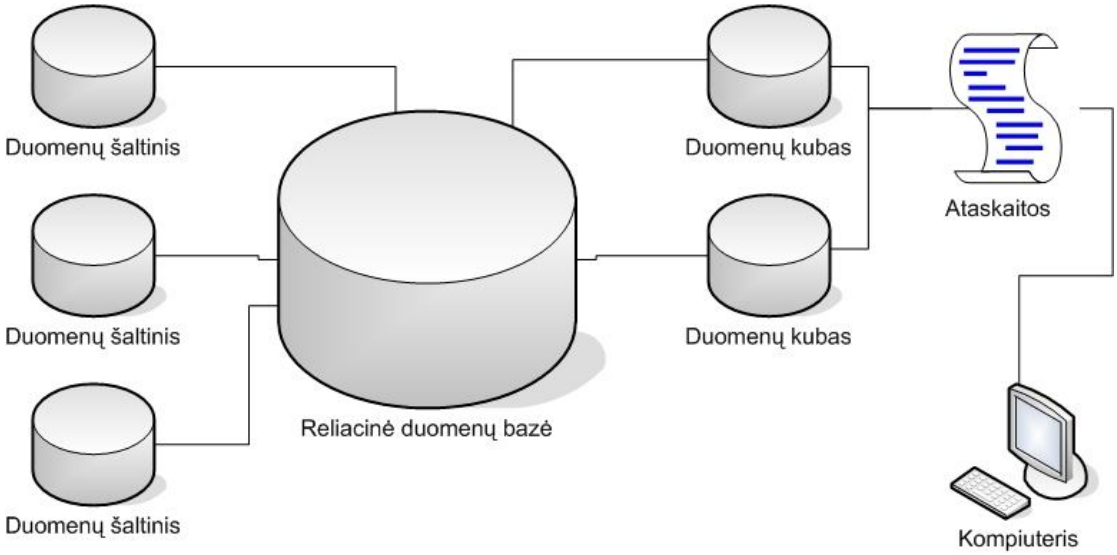
Sprendimų priėmimo sistemos dažniausiai turi agreguotų duomenų pateikimo vartotojui priemones, kurios suteikia galimybę iš pradinių duomenų rinkinio patogiai išrinkti analizei reikiamus duomenis. Dažniausiai tokios agreguotos funkcijos sudaro daugiamačius ir nereliacinius duomenų rinkinius, kurie vadinami hiperkubais arba metakubais. Šitų kubų ašys turi parametrus, o celės – nuo jų priklausančius agreguotus duomenis. Palei kiekvieną ašį duomenys gali būti organizuoti hierarchijos pavidalu, kuri parodo duomenų detalizacijos lygį. Tokio duomenų modelio dėka vartotojai gali formuluoti sudėtingas užklausas, generuoti ataskaitas, išgauti duomenis reikiamu pavidalu.

Kompleksinės daugiamatės duomenų analizės technologija gavo pavadinimą OLAP (*angl. On-Line Analytical Processing*). OLAP – tai vienas iš pagrindinių duomenų saugyklos organizavimo elementų. OLAP koncepciją aprašė 1993 metais Edgaras Kodas, kuris buvo žinomas duomenų bazių tyrinėtojas ir reliacinio duomenų modelio autorius. 1995 matais pagal Koddo reikalavimus buvo suformuluotas testas FASMI (*angl. Fast Analysis of Shared Multidimentional Information*), kuriame buvo pateikti reikalavimai daugiamatės analizės priedams.

- Per priimtiną (dažniausiai ne ilgiau 5 s.) laiką suteikti vartotojui analizės rezultatus. Tai turi būti padaryta netgi jeigu analizės detalumas bus mažesnis.
- Galimybė atlikti bet kokią loginę arba statistinę analizę ir jos rezultatus išsaugoti galutiniam vartotojui patogiu būdu.

- Kelių autorizuotų vartotojų priėjimas prie duomenų.
- Daugiamatis konceptualus duomenų pateikimas, įskaitant pilną hierarchijų ir daugialypių hierarchijų palaikymą.
- Galimybė prieiti prie bet kurios reikiamos informacijos, nepriklausomai nuo informacijos apimties ir jos saugojimo vietos.

Reikia pažymėti, kad OLAP funkcionalumas gali būti realizuotas skirtingais būdais, pradedant nuo paprasčiausių analizės priemonių ofiso įrankiuose ir baigiant paskirstytomis analitinėmis sistemomis, kurios naudojamos kartu su serverių produktais. Paveikslėlyje pav. 3 pavaizduota tipinės sistemos schema.



pav. 3. Tipinės OLAP sistemos schema

2.7.Techniniai daugiamačio duomenų saugojimo aspektai

Daugiamatėse duomenų saugyklose laikomi agreguoti skirtingo detalumo duomenis, pavyzdžiui, pardavimų apimtis dienomis, mėnesiais, metais, pagal prekių kategorijas ir t. t. Agreguotų duomenų saugojimo tikslas yra užklausų atlikimo greitis, todėl kad atliekant analizę ir prognozes, dažniausiai įdomūs ne detalūs, o suminiai duomenys. Todėl kuriant daugiamatę duomenų bazę, visada skaičiuojami ir saugojami agreguoti duomenis.

Pažymėsime, kad visų agreguotų duomenų išsaugojimas ne visada pasiteisina. Taip yra dėl to, kad pridedant naujas dimensijas duomenų, sudarančių kubą, apimtis auga pagal eksponentinį dėsnį (kartais sako „sprogstamas duomenų apimties augimas“). Jeigu kalbėti tiksliau, agreguotų duomenų apimties augimo laipsnis

priklauso nuo kubo dimensijų kaučiaus ir nuo dimensijų narių skirtinguose šių dimensijų hierarchijų lygiuose. Tam kad išspręsti „sprogstančio augimo“ problemą naudojamos skirtingos schemos, leidžiančios skaičiuojant agreguotus duomenis pasiekti priimtiną užklausų atlikimo greitį.

Kaip pradiniai, taip ir agreguoti duomenis gali būti saugomi arba reliacinėse arba daugiamatėse struktūrose. Todėl šiuo metų naudojami tris duomenų saugojimo būdai.

MOLAP (*angl. Multidinentional OLAP*) – pradiniai ir agreguoti duomenis saugomi daugiamatėje duomenų bazėje. Duomenų saugojimas daugiamatėse struktūrose leidžia manipuliuoti duomenimis kaip daugiamačiu masyvu, to dėka agreguotų reikšmių skaičiavimo laikas yra vienodas bet kuriai dimensijai. Bet šiuo atveju daugiamatė duomenų bazė tampa pertekline, nes daugiamačiai duomenys turi visus pradinius reliacinius duomenis [3].

ROLAP (*angl. Relational OLAP*) – pradiniai duomenis pasilieka toje pačioje reliacinėje duomenų bazėje, kurioje jie ir buvo. Agreguoti duomenis talpinami į specialiai jiems sukurtas tarnybines lenteles, toje pačioje duomenų bazėje [3].

HOLAP (*angl. Hybrid OLAP*) - pradiniai duomenis pasilieka toje pačioje reliacinėje duomenų bazėje, kurioje jie ir buvo, o agreguoti duomenis saugomi daugiamatėje duomenų bazėje [3].

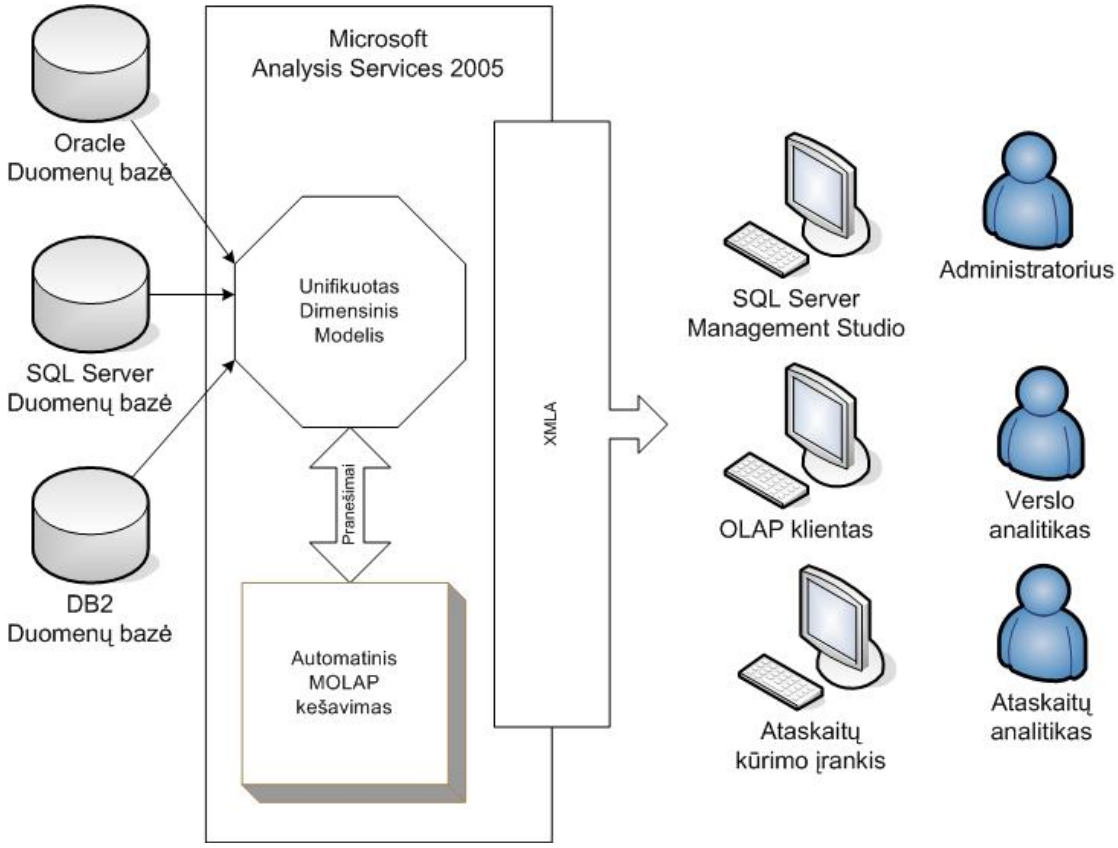
Kai kurie OLAP įrankiai palaiko duomenų saugojimą tik reliacinėse struktūrose, kai kurios tik daugiamatėse struktūrose. Bet dauguma šiuolaikinių serverinių OLAP įrankių palaiko visus tris duomenų saugojimo būdus. Duomenų saugojimo būdo pasirinkimas priklauso nuo pradinių duomenų kiekio ir apimties, užklausų atlikimo trukmės reikalavimų ir OLAP kubų atnaujinimo dažnumo.

2.8. Unifikuotas dimensinis modelis (UDM)

UDM suteikia galimybę sujungti prieigą prie heterogeninių duomenų šaltinių viename modelyje. UDM tai daugiau negu kubas, sudarytas iš kelių duomenų šaltinių. Jis apibrėžia kubus ir dimensijas ir leidžia atlikti tokius veiksmus, kaip papildomus skaičiavimus, duomenų vertimą, greitaveikos įvertinimą. UDM iš esmės duoda viskas kas yra geriausia OLAP ir RDMS technologijose. UDM duoda turtingus

metaduomenis, kurių reikia atliekant duomenų analizę, kartu su funkcionalumu, kuris leidžia atlikti skaičiavimus ir duomenų agregavimus kaip OLAP'e ir sudėtingas struktūras, galimybę naudotis patogiu užklausų kūrimo įrankiu, kurių reikia ataskaitų sudarymui. UDM gali ženkliai palengvinti kubų, turinčių daug faktų lentelių, kūrimą [8].

Paveikslėlyje pav. 4 pavaizduota bendra UDM schema, kurią realizuoja Microsoft Analysis Services 2005 įrankis.



pav. 4. Bendra UDM schema

Pagrindiniai UDM elementai yra:

- Heterogeninių duomenų prieigų palaikymas – UDM padeda integruoti ir suvienyti heterogeninius duomenų šaltinius, t.y. sujungti skirtingas duomenų struktūras į vieną unifikuotą modelį, kuris leidžią siųsti užklausas tik vienam modeliui.
- Spartus duomenų pasiekimas realiame laike – UDM sudaro MOLAP (*angl. multidimentional OLAP*) tipo kešą apjungiamiems duomenims. Kai atsiranda jungiamų bazės duomenų pakeitimai, MOLAP kešas sukuriamas iš naujo. Kai

vartotojas siunčia užklausas, jis gauna rezultatus iš MOLAP kešo. Kol vyksta MOLAP kešo atnaujinimas, duomenis imami iš reliacinės duomenų bazės.

- Išsamūs metaduomenys – UDM suteikia galimybę sudaryti apjungiamų duomenų šaltinių konsoliduotą vaizdą, ko pasėkoje galima gauti OLAP tipo metaduomenis. Dėl to, gautus metaduomenis galima naudoti duomenų peržiūrai, analizei, bei priimant sprendimus. UDM taip pat suteikia galimybę, priklausomai nuo analizės poreikių, peržiūrėti tam tikras unifikuoto modelio sekcijas.
- Gilios analizės palaikymas – kartu su turtingų metaduomenų palaikymų, UDM suteikia galimybę apibrėžti sudėtingus skaičiavimus, kuriuos galima pritaikyti apjungiamiems duomenims. Visi skaičiavimai, apibrėžti UDM viduje laikomi kaip esybė, kuri vadinasi MDX skriptai.
- Modelis ataskaitų kūrimui ir analizei – UDM duoda viską, kas yra geriausia OLAP ir RDMS technologijose. Jis suteikia ne tik galimybę kreiptis į agreguotus duomenis, kurie naudojami analizei, bet ir duoda galimybę daryti ataskaitas iki transakcijų, tarp heterogeninių duomenų šaltinių, detalumo

Dar vienas UDM patogumas yra tame, kad galima saugoti duomenų ir metaduomenų vertimus. Tai reškia, kad prisijungęs vartotojas, atitinkamai jo kompiuteryje nustatytos lokalizacijos, galės matyti informacija savo kalba.

2.9.Veiklos intelekto priemonės

Veiklos intelektas yra naudingų duomenų apie tam tikrą veiklą gavimo metodas, tiksliau metodų aibė. Veiklos intelekto funkcijas atlieka duomenų gavybos (*angl. Data Mining*) technologija. Ši technologija suteikia funkcijas, kurios leidžia aptikti duomenų saugykloje esančių duomenų dėsningumus, panašumus ir atlikti duomenų prognozavimą, klasifikavimą. Kodėl tai gali būti naudinga?

- Saugomų duomenų kiekiai auga labai greitai. Savaime aišku, kad visus duomenis išnagrinėti yra labai sudėtinga, o veiklos intelekto funkcijos gali aptikti nematomas duomenų savybes, kurios padės priimti teisinga verslo strategiją.

- Didelė konkurencija tarp organizacijų reikalauja ieškoti naujų būdų kaip išsilaikyti rinkoje, neprarasti klientų ir įgyti naujų. Veiklos intelektas gali aptikti duomenų savybes, kurios padės išspręsti tokius uždavinius.
- Šiuo metu tai yra pilnai subrendusi technologija, kuri naudoja pakankamai tikslius ir efektyvius algoritmus, kurie gali apdoroti labai didelius duomenų kiekius.

Su verslo intelekto funkcijomis galima išspręsti tokius uždavinius:

- Atsigręžimo analizė
- Pardavimų analizė
- Apgavysčių nustatymas
- Rizikos valdymas
- Klientų skaidymas
- Reklamos efektyvumo analizė
- Pardavimų prognozavimas

Tai yra tik keli uždavinių pavyzdžiai. Visus spręstinus uždavinių pagal sprendimo būdą galima sugrupuoti į kelias grupes ir apibrėžti universalius sprendimo metodus.

Klasifikavimas

Klasifikavimas priskiria tam tikrus požymius duomenų aibėms, turinčioms kažkokius bendrus atributus. Tokius atributus dažniausiai reikia aptikti, nes jie nėra akivaizdūs. Klasifikavimas gali būti atliekamas naudojant sprendimų medžių, Naive Bayers algoritmus ir neuroninius tinklus [11].

Grupavimas

Dar galima pavadinti kaip segmentavimą. Šis metodas naudojamas tada, kai pagal tam tikrus atributus reikia aptikti natūralias duomenų grupes. Visi įėjimo atributai apdorojami vienodai, ir dažniausiai ir tam, kad aptikti atributų bendrus požymius, reikia atlikti labai daug duomenų apdorojimo iteracijų, kurios stabdomos tada, kai stabilizuojasi grupių ribos [11].

Asociacija

Su šiuo metodu galima rasti dažnai kartu naudojamų objektų aibes. Pavyzdžiui, prekės, kurios dažniausiai patenka į vieną krepšelį. Dauguma algoritmų, su kuriais realizuojamas šis metodas, skenuoja duomenis kelis kartus, be to, dažniausiai dar reikia nurodyti dažnumo slenkstį ir bendrų objektų aibės dydį [11].

Regresija

Šis metodas yra panašus į klasifikavimą. Pagrindinis skirtumas yra tas, kad nuspėjamas atributas yra tolydus skaičius. Regresijos tipo uždaviniams spręsti dažniausiai naudojami linijinės ir loginės regresijos metodai, bet sprendimui taip pat galima naudoti regresijos medžius ir neuroninius tinklus [11].

Prognozavimas

Prognozavimas dažniausiai naudojamas pardavimų prognozėms daryti. Įėjimas dažniausiai būna laiko skalė (skaičių seka, kurie turi laiko prasmę). Tokių uždavinių sprendimui dažniausiai naudojamas ARIMA (*angl. AutoRegressive Integrated Moving Average*) metodas [11].

Sekų analizė

Sekų analizė naudojama tada, kai reikia diskrečių reikšmių aibėje rasti bendras savybes. Tai yra pakankamai nauja uždavinių aibė, kurių sprendimui galima pritaikyti Markovo grandinių metodą. Mokslininkai šiuo metu ieško naujų algoritmų, kurie būtu tinkami tokiems uždaviniams spręsti [11].

Nukrypimo analizė

Šis metodas taikomas tada, kai reikia rasti tokius objektus, kurie labai skiriasi nuo bendros objektų aibės. Šiuo metu nėra standartinių metodų nukrypimo analizei atlikti ir šitoje srityje vykdoma daug tyrimų. Dažniausiai analitikai naudoja sprendimų medžių grupavimo, neuroninių tinklų algoritmų modifikacijas [11].

Microsoft korporacija siūlo užklausų kalbą, kuri praplėsta veiklos intelekto funkcijomis. Ši kalba vadinasi DMX (*angl. Data Mining Extensions*). Ji naudojama duomenų išrinkimui, pritaikant uždavinio sprendimui tinkamiausią metodą, rezultatų įrašymui į lenteles ir duomenų apdorojimo šablonų sudarymui.

2.10. SQL, MDX ir DMX palyginimas

MDX tai yra kalba kuri leidžia išrinkinėti ir skaičiuoti duomenis iš OLAP duomenų bazės ir suteikia galimybę priduoti duomenims kažkokį vaizdų formatą. Priešingai, nei kitos OLAP kalbos, tai nėra pilnai ataskaitų formavimo kalba. MDX užklausos rezultatai grąžinami, kaip duomenų struktūra, kuri turi būti apdorota, taip, kad duomenis būtų vaizdūs ir suprantami. Tai panašu į tai, kaip SQL veikia su reliacinėmis duomenų bazėmis.

Iš pirmo žvilgsnio MDX sintaksė atrodo labai panaši į SQL sintaksę. Didžioji dalis funkcijų, kuria palaiko MDX, taip pat palaikomos ir SQL kalba. Su tam tikrom pastangom, su SQL kalba galima atkartoti MDX kalbai būdingą funkcionalumą [7].

Bet iš esmės tarp šitų kalbų yra labai didelis skirtumas jau konceptualiame lygyje. Esminis skirtumas yra tas, kad MDX skirtas dirbti su daugiamačiais duomenimis. Nors SQL Server Analysis Services 2005 leidžia dirbti su duomenų kubais naudojant SQL kalbą, MDX turi komandas, kurios specialiai skirtos darbui su dideliu dimensijų skaičiumi [7].

SQL gali dirbti tik su dviem dimensijom (stulpeliais ir eilutėmis) [10]. MDX, savo ruožtu, gali dirbti ir su trimis, keturioms ir daugiau dimensijomis. Dėl to dimensijos turi ašių prasmę. Eilutė ir stulpelis yra tik pirmų dviejų dimensijų ašių pavadinimai, kitų dimensijų pavadinimai neturi realios reikšmės, ir naudojami tik dėl atvaizdavimo.

MDX užklausos rezultatas yra kubas, kuris yra pirminio kubo transformacija. Tai yra panašu į SQL, todėl, kad SQL užklausa gražina lentelę. Grąžinamas kubas gali turėti iki 128 dimensijų, naudojant Analysis Services 2005, ir iki 64 – naudojant Analysis Services 2000 arba Essbase [5].

SQL kalboje SELECT komanda naudojama stulpelių formavimui, o WHERE – eilučių formavimui. MDX kalboje SELECT komanda gali būti panaudota kelių dimensijų išskirimui, o WHERE komanda naudojama tam, kad apriboti duomenų išrinkimą pagal tam tikras dimensijas.

SQL kalboje WHERE komanda naudojama tam, kad atfiltruoti gražinamus duomenis, bet MDX kalboje WHERE komanda naudojama, tam, kad gauti duomenų sluoksnį. Nors komandų koncepcija yra vienoda, bet rezultatas yra visiškai skirtingas. SQL užklausos kūrimo procesas taip pas skiriasi nuo MDX užklausos kūrimo proceso.

SQL užklausos rezultatas vizualizuojamas gana intuityviai, kaip vienos arba kelių sujungtų lentelių eilučių rinkinys (paprasta lentelė). Kadangi MDX užklausos rezultatas gali turėti daugiau, nei tris dimensijas, jos rezultato atvaizdavimas nėra toks intuityvus ir gali būti panašus į struktūrą [7].

Pradėsime nagrinėti paprasčiausią MDX užklausą, pridedami prie jos papildomus blokus. Taip lengvai galima bus suprasti, kaip veikia tokios užklausos. Tarkime turime paprastą kubą su laiko, vietovių ir pardavimo matų dimensijomis. Kubą pavadinsime Sales. Paimsime skaičių lentelę, kuri parodo pardavimus Vilniuje

už 2006 metų pirmą ir antrą ketvirčius. Rezultatus gausime paprastoje lentelėje, o paprasta lentelė gali turėti tik dvi dimensijas. Užklausa gali atrodyti taip:

MDX

```
SELECT
{ [Measures].[Dollar Sales], [Measures].[Unit Sales] }
on columns,
{ [Time].[Q1, 2005], [Time].[Q2, 2005] }
on rows
FROM [Sales]
WHERE ([Customer].[MA])
```

SQL

```
SELECT
    T.quarter, SUM(M.[ Dollar Sales]) , SUM(M.[Unit Sales])
FROM
    Measures M, Time T, Customer C
WHERE
    M.time_id = T.time_id and
    M.cust_id = C.cust_id
    T.year = 2006 and
    C.name = 'MA'
GROUP BY
    T. quarter
```

Rezultatas gali atrodyti taip:

MDX

	Pajamos	Kiekis
Q1, 2006	1495632,22	39456
Q2, 2006	2457611,98	45832

arba

SQL

Ketvirtis	Pajamos	Kiekis
1	1495632,22	39456
2	2457611,98	45832

Matome, kad SQL užklausoje reikia papildomai daryti lentelių sujungimus, grupavimą ir sumavimą. Nors išrinkimas yra elementarus, o skirtumas tarp užklausų akivaizdus.

DMX užklausos yra praktiškai identiškos SQL užklausoms. Vienas iš pagrindinių skirtumų yra tas, kad FROM bloke nurodom ne lentelę o duomenų gavybos šabloną. Elementariausias pavyzdys gali atrodyti taip:

```
SELECT
      T.[quarter], M.[Dollar Sales], M.[Unit Sales]
FROM
    Measures M PREDICTION JOIN
        OPENQUERY ([BD], ,SELECT * FROM Time WHERE [year] = 2006') T
      ON M.[tyme_id] = T.[time_id]
WHERE
      M.[Dollar Sales] > 1000
```

Čia *Measures* yra duomenų gavybos šablonas, kuris jungiamas ne su lentele, o su paprastos SQL užklausos rezultatais.

Komandos SELECT, FROM, WHERE užklausą padalina į atskiras dalis. MDX užklausos rezultatas yra lentelę, iš esmės naujas kubas. Dimensijos tiesiog priskiriamos prie rezultatų lentelės ašių (eilutėms ir stulpeliams). MDX terminologijoje, ašis yra viena iš užklausos rezultato dimensijų. Kiekviena ašis gali turėti skirtingų kubo dimensijų kombinacijas. Panagrinėsime užklausą dalimis [1]:

- Užklausa pradedama komanda SELECT, kuri leidžia nurodyti, ką norime išgauti.
- Komanda ON naudojama kartu su ašies pavadinimu, kuris parodo, kur turi patekti dimensijos. Pavyzdžiui, galima padaryti taip, kad pardavimo matų dimensija būtų parodyta kaip stulpeliai, o laiko dimensija – kaip eilutės.
- MDX užklausose naudojami skliausteliai { ir }, tam, kad suvienyti dimensijos arba kelių dimensijų elementus. Elementai atskiriami kableliais, o jų vardai rašomi laužtiniuose skliaustose []. Be to elementas nurodomas taip: [Dimensija].[Elementas].
- MDX užklausoje pasirenkamos dimensijas, kurios patenka ant rezultatų ašių. Užklausa gali turėti skirtingą ašių kiekį. Pirmos trys ašys vadinamos stulpeliai, eilutės ir puslapiai, kad sutampa su spausdintinės ataskaitos koncepcija.
- FROM komanda leidžia pasirinkti kubą, iš kurio imsime duomenis.
- WHERE komanda leidžia nurodyti apribojimus pagal dimensijas, kurios nebuvo nurodytos nei prie vienos ašies. Ši komanda nėra būtina.

Nors SELECT, FROM, WHERE MDX užklausos konstrukcija yra labai panaši į SQL užklausos konstrukciją, bet reikšmė ir semantika yra nevisai vienoda.

MDX užklausose galima naudoti daug ašių. Pirmos penkios ašys turi tokius pavadinimus: 0 – Columns, 1 – Rows, 2 – Pages, 3 – Chapters, 4 – Sections. Jeigu reikia naudoti daugiau ašių, reikėtų rašyti ašies numerį, pavyzdžiui:

```
SELECT
    {[Customer].[MA], [Customer].[CT] }
on axis(0),
    { [Time].[Q1, 2005], [Time].[Q2, 2005], [Time].[Q3, 2005] }
on axis(1)
FROM
    Sales
WHERE
    ([Measures].[Dollar Sales])
```

Užklausoje galima naudoti kartu ir ašių pavadinimus ir ašių numerius. Reikia pažymėti, kad jeigu užklausoje naudojama n-toji ašis, tai būtinai turi būti panaudoto visos ašys iki n = 0.

MDX kalboje galima naudoti kortežus ir rinkinius.

Kortežas yra vienos arba daugiau dimensijų narių kombinacija. Tai yra labai svarbus daugiamačių užklausų narys. Vienas narys yra paprastas kortežas (pavyzdžiui `[Time].[Jun, 2005]`). Kai kortežas turi daugiau nei vieną dimensiją, jis turi po viena kiekvienos dimensijos narį. Tam kad skirtingų dimensijų narius įkelti į vieną kortežą, reikia juos užrašyti vieną po kito (pavyzdžiui `([Customer].[Chicago, IL], [Time].[Jan, 2005])`). [1]

Kai kubas dalinas į kiekvieną kortežo narį, kortežas gali būti kubo sluoksniu. Kortežas taip pat gali būti atskiru duomenų tipų, kurio galima operuoti atskirai.

```
SELECT
    { ( [Time].[2005], [Measures].[Dollar Sales] ),
    ( [Time].[Feb, 2005], [Measures].[Unit Sales] )
    }
ON COLUMNS ,
    { [Product].[Tools], [Product].[Toys]} ON ROWS
FROM [Sales]
```

	2005	Feb, 2005
	Dollar Sales	Unit Sales
Tools	1,083,855.90	2,621
Toys	848,982.70	1,695

Šiame pavyzdyje gavome asimetrinę laiko ir matų dimensijų kombinaciją. Galimybė taip padaryti duoda didelį lankstumą duomenų išrinkimui. Užklausoje negalima naudoti tuščių kortežų.

Rinkinys yra paprastas kortežų rinkinys. Vienas rinkinys gali turėti daugiau nei vieną kortežą. Be to, vienas rinkinis gali turėti kelis vienus kortežus.

```
SELECT
    { ( [Time].[2005], [Measures].[Dollar Sales] ),
    ( [Time].[Feb, 2005], [Measures].[Unit Sales] )
}
ON COLUMNS ,
    { [Product].[Tools], [Product].[Toys] } ON ROWS
FROM [Sales]
```

Aukščiau pateiktame pavyzdyje užklausa turi du rinkinius. Stulpeliai turi vienos dimensijos rinkinį, o eilutės turi dviejų dimensijų rinkinį. Rinkinius reikia rašyti skliaustose. Kai kurios MDX funkcijos ir operatoriai taip pat grąžina rinkinius. Nors vienas narys pagal nutylėjimą yra vienos dimensijos kortežas, bet rinkinis, turintis vieną kortežą nėra rinkinys.

Taigi trijų užklausų kalbos labai skiriasi, nes skiriasi jų aplinka. SQL geriausiai dirba su vienmačiais duomenimis. MDX dirba tik su daugiamačiais

duomenimis. DMX gali dirbti ir su vienmačiais ir su daugiamačiais duomenimis. Taip pat su šitų kalbų pagalba galima kurti naujas struktūras. SQL kalba leidžia sukurti duomenų lentelę, MDX kalba leidžia sukurti duomenų kubą, o DMX kalba leidžia sukurti duomenų apdorojimo šabloną. Taigi kalbų galimybių palygimas pateikiamas lentelėje 1.

lentelė 1. Užklausų kalbų galimybės

	DML	**DDL**
SQL	+	+
MDX	+	+
DMX	+	+

2.11. Analizės išvados

SQL ir MDX analizė ir palyginimas parodė, kad tai tikrai yra skirtingos kalbos, kurios gali pasirodyti panašios tik iš pirmo žvilgsnio. Iš esmės tai yra skirtingos kalbos. SQL geriau nenaudoti darbui su daugiamačiais duomenimis, nes tai yra neefektyvu, neoptimalu ir nepatogu, na o MDX galima taikyti tik darbui su daugiamačiais duomenimis, todėl jis taip pralenkia SQL užklausų kalbą šitoj srity.

Kitas dalykas yra DMX kalbos šablonų naudojimas. Standartizavus duomenų šaltinius ir apibrėžus šaltinio sritį, galima nagrinėti ir DMX užklausų šablonų naudojimą. Duomenų šaltinių standartizavimas reikalingas tam, kad būtų aišku kokio pobūdžio ir kokios konstrukcijos užklausų šablonus kurti. Jeigu neaiški sritis ir reikalavimai, tai neaiški ir konstrukcija.

3. Daugiamačių duomenų prognozavimo metodas taikant DMX šablonus

Duomenų prognozavimas - yra naujų duomenų sukūrimas remiantis sukauptais duomenimis. Realūs, per tam tikrą laikotarpį, surinkti duomenis gali turėti iš pirmo žvilgsnio nematomus dėsningumus, savybes ir požymius, pagal kuriuos su tam tikrą paklaidą galima būtų nustatyti būsimus duomenys. Tai galėtų būti prekių pardavimo analizė, demografinės aplinkos tyrimas, gamtos rodyklių analizė ir kitos sritys, kuriose tokie duomenys gali būti naudingi. Sudėtingiausias uždavinys yra rasti apdorojamų duomenų dėsningumus ir požymius, kuriais remiantis galima būtų daryti prognozės.

3.1. Metodo tikslas

Automatizuoti trūkstamų saugyklos duomenų nustatymą ir generavimą, sukuriant DMX šablonus.

3.2. Funkciniai ir nefunkciniai reikalavimai metodui

Nefunkciniai reikalavimai

Duomenų prognozavimo metodas taikomas veiklos duomenų prognozavimui. Veiklos duomenis turi būti laikomi duomenų saugykloje .Prognozavimui atlikti naudojamas duomenų gavybos šablonas. Labai svarbu, kad duomenų saugykloje būtų pakankamas kiekis pradinių duomenų, pagal kuriuos galima būtų daryti prognozes. Kadangi prognozavimas vykdomas laiko atžvilgiu, būtina turėti duomenis už kelis veiklos periodus, kiekvienas iš kurių turi būti ne trumpesnis už prognozuojamą veiklos periodą.

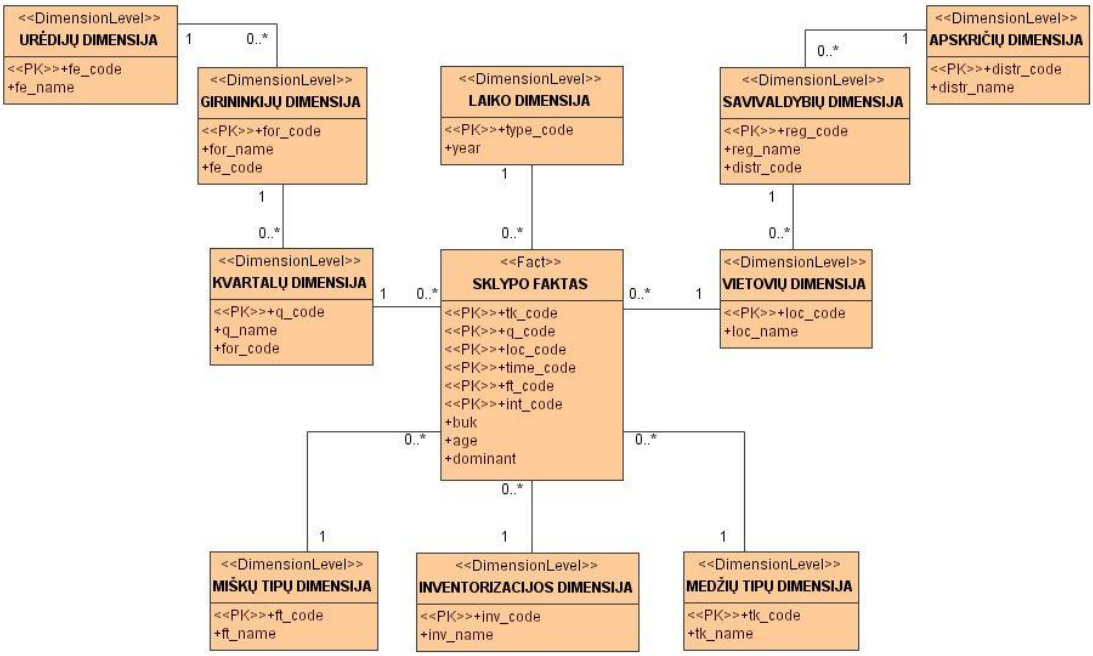
Funkciniai reikalavimai

Veiklos prognozė galima išskirti į keturias stambias funkcijas: prognozuojamo atvejo nustatymas, pradinių duomenų paruošimas, pradinių duomenų įkrovimas, prognozavimo šablono naudojimas, rezultatų transformavimas ir įkrovimas. Visas funkcijas inicijuoja duomenų saugyklos administratorius.

3.3.Metodo eskizas

Pagal nefunkcinius reikalavimus mums reikia sudaryti duomenų saugyklą, kurioje būtų laikomi veiklos duomenys. Kadangi saugykla yra duomenų šaltinis, reiškia tai yra pagrindinis elementas nuo kurio reikia pradėti modelio sudarymą. Saugyklos modelis apibrėš modelio struktūrą ir sumažins abstrakcijos lygį.

Sudarysime saugyklos modelį, kuris apibrėžia Lietuvos sklypų, kuriuose auga miškai, struktūrą. Duomenų saugyklos modelis pavaizduotas paveikslėlyje pav. 5.

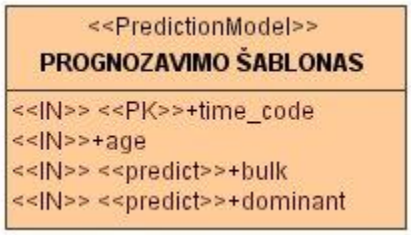


pav. 5. Duomenų saugyklos modelis

Tokia saugykla turi klasikinės snaigės formą. Faktų lentelėje saugoma informacija apie sklypuose augančius medžius. Ši informacija detalizuojama pagal laiką (metais), inventorizacijos periodą, urėdijas, girininkijas, sklypus, miškų tipus, medžių tipus, apskritis, savivaldybes ir vietoves. Tarkime kad saugykla papildoma duomenimis kas penkis metus ir juos analizuojant daromos išvados, kur galima kirsti medžius. Penki metai yra pakankamai ilgas laiko tarpas, tuo labiau, kad realūs duomenis keičiasi žymiai greičiau. Todėl atsiranda naujos informacijos trūkumas. Šiai sričiai pritaikius prognozavimo metodą, galima būtų išspręsti duomenų trūkumo problemą.

Kai saugyklos struktūrinis modelis yra sudarytas, galima apibrėžti prognozavimo šablono modelį. Kadangi prognozavimas turi vykti laiko atžvilgiu,

reiškia duomenų saugyklos laiko dimensija turės rakto atributą. Prognozuojami laukai yra medžio amžius, aukštis, tūris. Tokio šablono struktūra pavaizduota paveikslėlyje pav. 6. Medžių augimo prognozavimo šablonas turėtų išanalizuoti medžių augimo priklausomybę nuo laiko, rasti augimo koeficientus ir pagal juos atlikti prognozę.

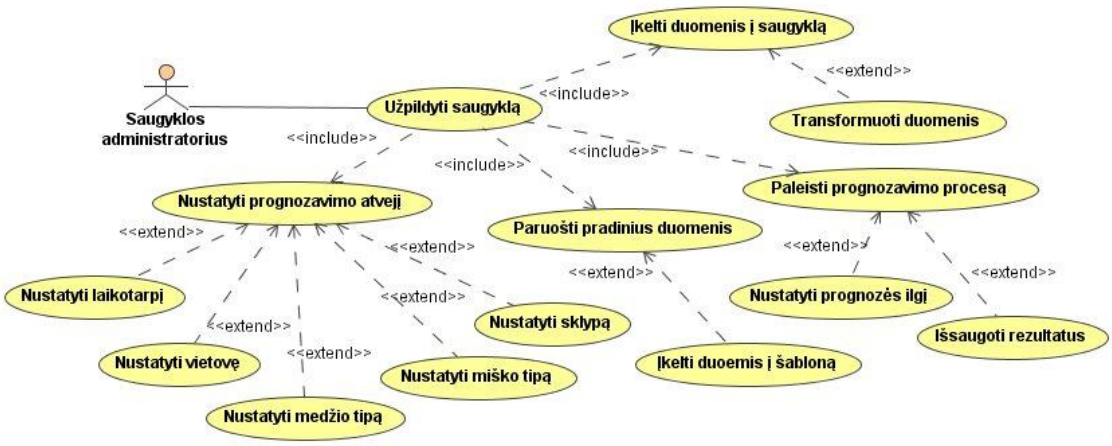


pav. 6. Medžių augimo prognozės šablonas

Šio šablono struktūra sudarinėjama pagal duomenų saugyklos modelį. Labai svarbu sudaryti teisingą šablono modelį, nes nuo to gali priklausyti jo parodomi rezultatai. Šablono sudarymo procesą galima išskaidyti į tris žingsnius:

- Tuščio šablono sudarymas – tai šablono įėjimų ir išėjimų apibrėžimas. Šitas procesas labai panašus į paprastos lentelės sukūrimą, reikia nurodyti įėjimo laukus ir jų tipus. Taip pat reikia pažymėti laukus, kuriems ir lauką, pagal kuriuos atliekama prognozė.
- Šablono derinimas – šio žingsnio metu į sudarytą šabloną užkrauname duomenis. Po to vykdomas šablono derinimas, paduodant į šabloną duomenų sluoksnius ir nagrinėjant jo atrastas duomenų priklausomybes. Tai pakankamais ilgas procesas kuris atliekamas daug kartų ir reikalaujantis kruopštumo.
- Šablono testavimas – šio žingsnio metu į suderintą šabloną paduodama nauja informacija ir nagrinėjami rezultatai.

Kai duomenų saugyklos ir prognozavimo šablono modeliai sudaryti, galima detalizuoti viso metodo funkcinį modelį. Tai reiškia, kad galima apibrėžti funkcijas, su kurių pagalba bus atliekamas automatinis saugyklos užpildymas trūkstamais duomenimis. Funkcinis modelis pavaizduotas paveikslėlyje pav. 7.



pav. 7. Funkcinių reikalavimų modelis

Funkcinį modelį sudaro keturios pagrindinės funkcijos: nustatyti prognozavimo atvejį, paruošti pradinius duomenis, paleisti prognozavimo procesą, įkelti duomenis į saugyklą. Toliau aprašysime kiekvieną funkciją smulkiau.

Nustatyti prognozavimo atvejį – ši funkcija turi paruošti duomenų filtrą, kuris atfiltruotų užduoto laikotarpio informaciją apie vieną sklypą. Kadangi šablonas turi tik laiko dimensija, tai užkrovus į šabloną visą informaciją be apribojimų, negausime reikiamo detalumo. Todėl šablonas turi naudoti tik reikiamo detalumo duomenis. Šioje funkcijoje atliekami tokie veiksmai: laikotarpio nustatymas (kiek istorinių duomenų naudosime analizėje), vietovės nustatymas, medžio tipo nustatymas, miško tipo nustatymas, sklypo nustatymas. Visi šitie veiksmai iš esmės yra visų dimensijų reikiamų reikšmių surinkimas.

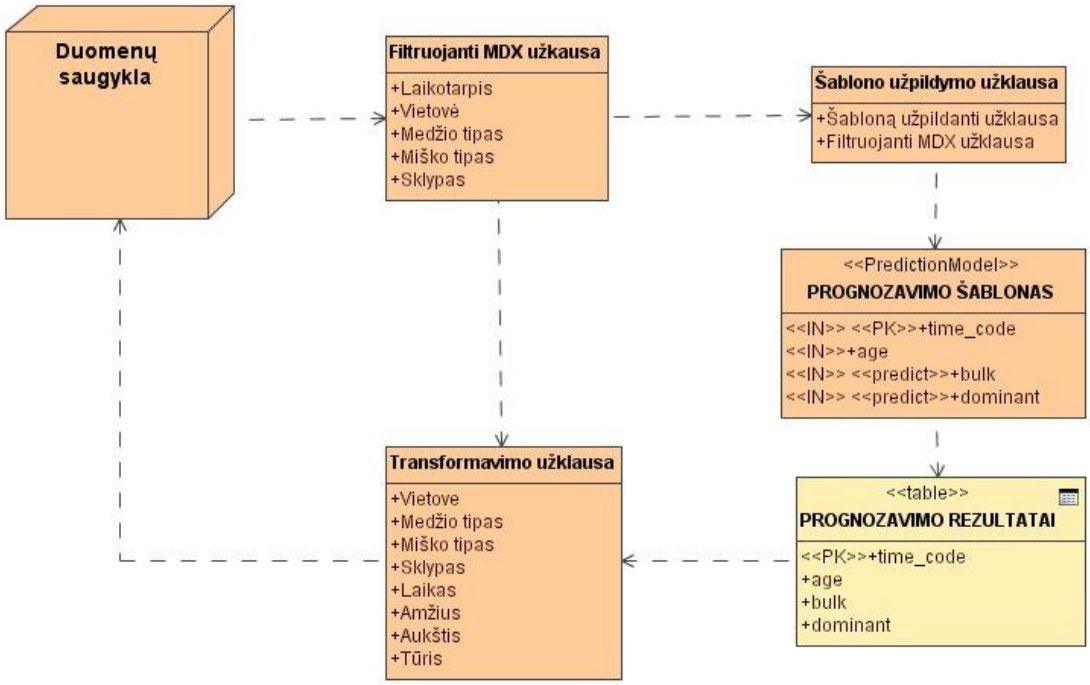
Paruošti pradinius duomenis – ši funkcija turi paimti pirmoje funkcijoje sudarytą filtrą, įdėti jį į šablono papildymo užklausą. Kai šablono papildymo užklausa paruošta, vykdomas duomenų įkėlimas į šabloną.

Paleisti prognozavimo procesą – ši funkcija turi inicijuoti šablono paleidimą. Šablonas turi būti paleistas su papildomu parametru, kuris apibrėžia prognozavimo laikotarpį. Tai turi atlikti papildoma funkcija „Nustatyti prognozės ilgį“. Kai šablonas grąžina rezultatus, papildoma funkcija „Išsaugoti rezultatus“ turi įrašyti rezultatus į atskirą struktūrą (pavyzdžiui lentelę).

Įkelti duomenis į saugyklą – ši funkcija turi paimti duomenis iš atskiros struktūros ir įrašyti juos į duomenų saugyklą. Prieš įrašymą į duomenų saugyklą duomenis reikia transformuoti ir suteikti jiems pilnai atitinkanti duomenų saugyklos faktų lentelės formatą. Tai būtina atlikti todėl kad, šablono grąžinami duomenis turi

tik laiko dimensijos kodą ir suprognozuotus faktinius skaičius. Visų kitų dimensijų kodus reikia suteikti transformacijos metu.

Tam, kad duomenų saugyklos užpildymo procesas būtų suprantamas ir vaizdus reikia sudaryti jo modelį. Paveikslėlyje pav. 8 pavaizduotas duomenų saugyklos užpildymo, panaudojant duomenų prognozavimo šabloną, proceso modelis.



pav. 8. Duomenų saugyklos užpildymo proceso modelis

Ši diagrama parodo kas ir kokia tvarka dalyvauja saugyklos užpildymo procese. Pirmiausiai generuojama MDX užklausa, kuri naudojama kaip filtras reikiamam duomenų kubo sluoksniui gauti. Filtruojanti MDX užklausa gali būti šabloninė, nes joje turėtų keistis tik dimensijų reikšmės (laikotarpis, vietovė, medžio tipas, miško tipas, sklypas). Toliau formuojama prognozavimo šablono užpildymo užklausa, kuri susideda ir filtruojančios MDX užklausos ir užpildymo užklausos. Ji taip pat gali būti šabloninė, nes joje gali keistis tik filtruojanti MDX užklausa. Kai šablonas paruoštas, jis atlieka skaičiavimus ir rezultatus patalpina į lentelę, kurios formatas turi būti suderintas su šablono grąžinamų rezultatų formatu. Po to suformuojama transformavimo užklausa. Ši užklausa formuojama pridedant prie suprognozuotų duomenų formato, filtruojančios MDX užklausos parametrus, taip suteikiant duomenims saugyklos faktų lentelės formatą. Tokio formato duomenys

įrašomi į saugyklą. Transformuojanti užklausa taip pat gali būti šabloninė, nes duomenų formatas visada išlieka atstovus.

Reikia pažymėti, kad toks užpildymo procesas turi būti atliekamas ciklišk ai, kiekviena kartą filtruojančiai MDX užklausai paduodant naujo sklypo dimensijų reikšmes.

3.4.Prognozavimo šablono veikimo principas

Šame skyrelyje aprašysime duomenų prognozavimui naudojamo šablono veikimo principą. Tai turėtų duoti daugiau supratimo apie tai kas yra duomenų prognozavimo šablonas ir kokius algoritmus jis naudoja.

Duomenų prognozavimo šablonas naudoja Microsoft Time Series algoritmą. Tai yra originalus prognozavimo algoritmas, kuris iš esmės yra autoregresijos ir sprendimų medžių algoritmų derinys, kitaip dar vadinamas ART (*angl. Auto Regression Tree*) algoritmu. Autoregresijos algoritmas dažnai naudojamas sprendžiant uždavinius su laiko eilutėmis.

Pirmiausiai reikia apibrėžti kintamųjų seką $Y = (Y_1, Y_2, \ldots, Y_T)$. Laiko eilučių duomenys yra reikšmių seka, kuri atitinka $y = (y_1, y_2, \ldots, y_T)$ kintamuosius. Toliau galima padaryti prielaidą, kad nagrinėjamas laiko eilučių modelis yra nesikečiantis, tikimybinis ir turi Markovo p-eiliškumą ($p \geq 0$). Tokį modelį galima aprašyti kaip

$$p(y_t|y_1, \ldots, t_{t-1}, \theta) = f\left(y_t|y_{t-p}, \ldots, y_{t-1}, \theta\right), p < t \leq T$$

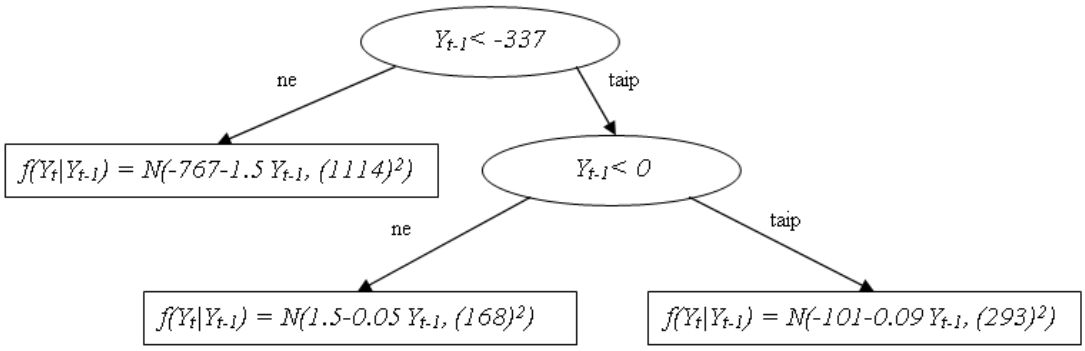
Čia $f(.|., \theta)$, yra sąlyginių tikimybių paskirstymų šeima, kuri parodo funkcinį modelio pavidalą, o Θ yra modelio parametrai. Sakydami, kad modelis yra nesikeičiantis, turime omenyje tai, kad y_t priklausomybė nuo anksčiau einančių kintamųjų laikui bėgant nesikeičia. Prielaida apie Markovo p-eiliškumą reiškia tai, kad praeiti p žingsniai y_t nepriklauso nuo kitų praeitų žingsnių. Reikia pažymėti tai, kad funkcija $f\left(y_t|y_{t-p}, \ldots, y_{t-1}, \theta\right)$ dažnai dar vadinama regresija, kur Y_t yra planuojamas kintamasis, o $(Y_{t-p}, \ldots, Y_{t-1})$ yra regresijos kintamieji.

Dažniausiai laiko eilučių analizėje naudojamas tiesinis autoregresinis modelis. Ilgio p tiesinį autoregresinį modelį, kuris žymimas kaip *AR(p)*, galima aprašyti taip:

$$f\left(y_t|y_{t-p}, \ldots, y_{t-1}, \theta\right) = N\left(m + \sum_{j=1}^{p} b_j y_{t-j}, \sigma^2\right)$$

Čia $f(y_t|y_{t-p},\dots,y_{t-1},\theta)$ yra tiesinė regresija, $N(\mu,\sigma^2)$ yra normalus pasiskirstymas, kurio vidurkis yra μ ir laisvumo laipsnis σ^2, o $\theta=(m,b_1,\dots,b_p,\sigma^2)$ yra modelio parametrai.

Autoregresijos medžio ART modelis atvirkštinis tiesinės regresijos modelis, kuriame ribas nustato sprendimo medis, o sprendimo medžio lapai turi tiesinius autoregresinius modelius. Paveiksle 9 pavaizduotas *ART* modelis kuris turi tris lapus. Kiekvienas lapas turi *AR(1)* modelio apibrėžimą.



pav. 9. Autoregresijos medis

Toliau bus nagrinėjami modeliai, kurie vadinami ilgio *p* autoregresijos medžio modeliais ir žymimi kaip *ART(p)*. *ART(p)* modelyje kiekvienas sprendimų medžio lapas turi *AR(p)* modelį, o sprendimų medžio skaidymo kintamieji parenkami iš praeitų *p* reikšmių laiko eilutėje.

ART(p) modelyje kiekvienas sprendimų medžio mazgas turi su juo susietą loginę formulę, kuri yra p kintamųjų funkcija Y_{t-p}, ..., Y_{t-1}. Pavyzdžiui pradinis ART modelio, kuris pavaizduotas paveiksle 9, mazgas tikrina, ar Y_{t-1}<*-337*. Kiekvienas sprendimų medžio lapas l_i, turi jam priskirtą formulę φ_i, kuri grąžina 1 kai visos formulės nuo bazinio mazgo iki mazgo l_i grąžina teigiamą loginę reikšmę, ir grąžina 0 priešingu atveju. Taigi *ART(p)* modelį galima aprašyti taip:

$$f(y_t|y_{t-p},\dots,y_{t-1},\theta)$$
$$=\prod_{i=1}^{L} f_i(y_t|y_{t-p},\dots,y_{t-1},\theta_i)^{\phi_i}=\prod_{i=1}^{L} N(m_i+\sum_{j=1}^{p} b_{ij}y_{t-j},\sigma_i^2)^{\phi_i}$$

Čia L yra sprendimų medžio lapų skaičius, $\Theta=(\Theta_1,\dots,\Theta_L)$ ir $\Theta_i=(m_i,b_{i1},\dots,b_{ip},\sigma_i^2)$ modelio parametrai tiesiniai regresijai l_i lape, kur *i = 1, ..., L*.

ART(p) (ir *ART*) modeliai yra *AR* modelių apibendrinimas, nes tik vieną lapą turintis *ART(p)* modelis iš esmės yra *AR(p)* modelis. *ART(p)* modeliai yra galingesni nei *AR* modeliai tuo, kad jie gali modeliuoti netiesinius sąryšius laiko eilučių

duomenyse. Be to, *ART(p)* modeliai gali pavaizduoti periodiškumą laiko eilučių duomenyse.

Toliau bus apžvelgtas Bayesian metodas modelių apmokymui. Remiantis šiuo metodu, turime modelių aibę $s_1, \ldots, s_S$, kuri turi modelių parametrų aibę $\Theta_{s1}, \ldots, \Theta_{sS}$, atitinkamai, ir galia suteikti modelio struktūros ir parametrų suderinamumą parodančius koeficientus, priskirdami tikimybinius pasiskirstymus $p(s)$ ir $p(\Theta_s \mid s)$. Po to galia naudoti Bayes taisyklę kartu su duomenimis d išreiškiant vėlesnius pasiskirstymus per $p(s \mid d)$ ir $p(\Theta \mid d, s)$ reikšmes. Bendru atveju, po to galima daryti prognozavimą naudojant suvidurkintus pasiskirstymus. Bet šio metodo taikymo metu naudojamus skaičiavimus atsekti praktiškai neįmanoma. Dėl tos priežasties dažniausiai naudojamas Baysean modelio pasirinkimo metodas, kuris leidžia pasirinkti struktūrą s, kuri turi didžiausią istorinę tikimybę $p(s \mid d)$.

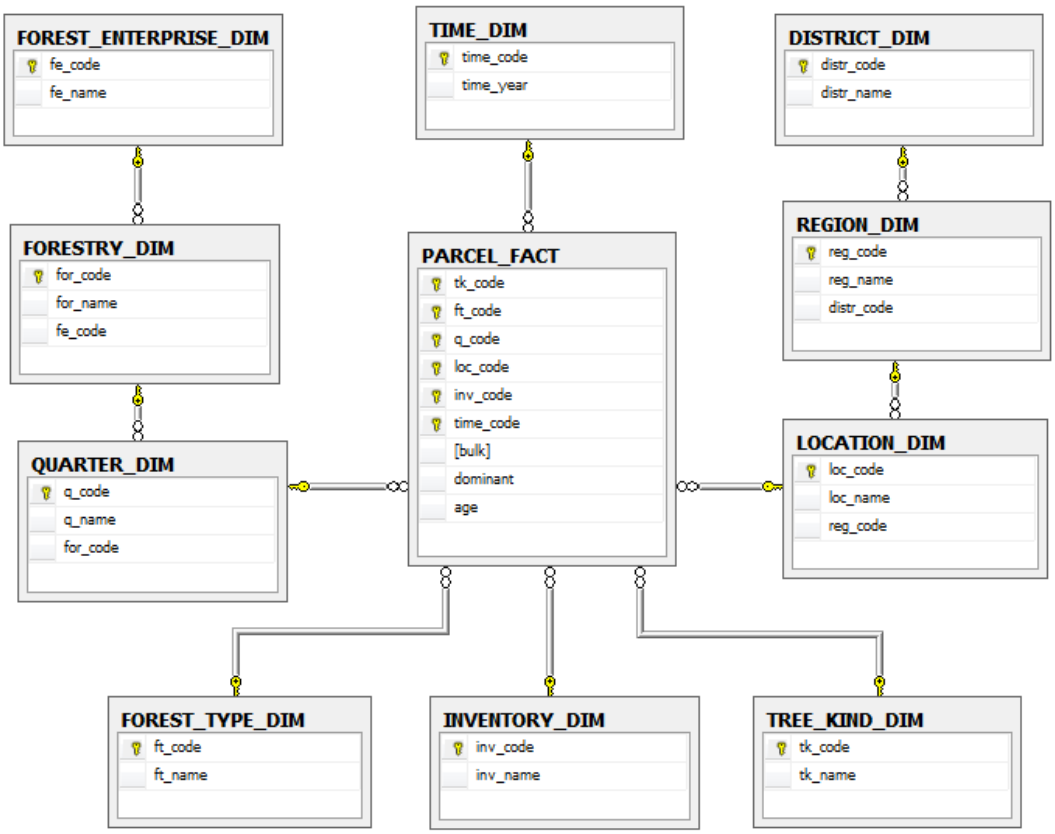
Taigi galima padaryti išvada, kad ART modeliai yra naudingi laiko eilučių analizėje, o Bayesian metodas, kuris taikomas ART modelių apmokymui, leidžia pasirinkti tiksliausius prognozavimo rezultatus parodančius modelius. Be to, naudojant modelio pasirinkimo metodą, ART modelių rezultatus įmanoma atsekti. ART modeliai taip pat suteikia paprastai interpretuojamą alternatyvą kitiems netiesiniams metodams, pavyzdžiui neuroniniams tinklams [12].

4. Metodo taikymas eksperimentinei sistemai

Šiame skyriuje aprašomas anksčiau aprašytai probleminei sričiai atliktą duomenų prognozavimo eksperimentą. Eksperimentinei sistemai realizuoti buvo pasirinktas Microsoft SQL Server 2005 paketas. Šis paketas yra vienas populiariausių, nes turi labai daug galimybių, yra stabilus, patogus naudoti ir padarytas pagal principą daug viename. Microsoft SQL Server 2005 paketą sudaro reliacinių duomenų bazių modulis, veiklos intelekto įrankių modulis, duomenų integracijos modulis, duomenų atvaizdavimo (ataskaitų) modulis ir pranešimų modulis. Tai yra labai patogu, nes visi naudojami įrankiai yra maksimaliai unifikuoti, turi vienodą valdymo logika ir gerą tarpusavio integraciją. Reikia pažymėti tai, kad eksperimentinei sistemai sudaryti reikia galingiausios sistemos versijos Microsoft SQL Server Enterprise Edition, nes kitose versijoje yra labai ribojamos duomenų gavybos funkcijų naudojimas. Dėl to yra būtina naudoti Microsoft Windows Server operacinę sistemą, nes galingiausia SQL Server versija yra suderinama tik su Server serijos operacinėmis sistemomis.

4.1.Eksperimentinė duomenų saugykla

Eksperimentui atlikti reikia realios duomenų saugyklos ir duomenų. Realizuotos duomenų saugyklos schema pavaizduota paveikslėlyje pav. 10. Ši duomenų saugykla atitinka anksčiau aprašytos saugyklos modelį. Joje laikomi detalūs duomenys apie miškus, t.y. informacija apie medžių tūrį ir aukšti su detalizacija pagal metus, inventorizacijos metus, vietovę (taip pat pagal rajoną ir apskritį), medžių tipą, miško tipą ir tikslią vietą (taip pat pagal girininkiją ir urėdiją). Detalus duomenų saugyklos aprašymas pateiktas lentelėje 2. Tokios struktūros duomenų saugyklos pagrindu buvo sukurtas OLAP duomenų kubas. Duomenų saugykla naudojama kaip duomenų kubo šaltinis. Toliau aprašytas prognozavimo šablonas prijungiamas prie OLAP kubo ir naudoja jame esančius duomenis prognozėms atlikti.



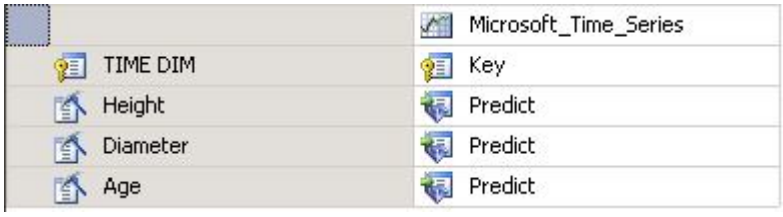
pav. 10. Duomenų saugyklos schema

lentelė 2. Duomenų saugyklos aprašymas

Lentelė	Aprašymas
PARCEL_FACT	Pagrindinė faktų lentelė, kurioje saugojami duomenis apie sklype augančius medžius (tūris, aukštis, amžius).
TIME_DIM	Detalizacijos pagal laiką dimensija (detalizacijos lygis - metai).
DISTRICT_DIM	Detalizacijos pagal apskritį dimensija.
REGION_DIM	Detalizacijos pagal rajoną dimensija.
LOCATION_DIM	Detalizacijos pagal vietovę dimensija.
TREE_KIND_DIM	Detalizacijos pagal medžių tipą dimensija.
FORST_TYPE_DIM	Detalizacijos pagal miško tipą dimensija.
INVENTORY_DIM	Detalizacija pagal inventorizacijos metus (metai, kada buvo atliktas tikrų duomenų surinkimas).
QUARTER_DIM	Detalizacijos pagal sklypo vietą dimensija.
FORESTRY_DIM	Detalizacijos pagal girininkiją dimensija.
FORES_ENTERPRISE_DIM	Detalizacijos pagal urėdiją dimensija.

4.2.Prognozavimo šablono struktūra

Pagal anksčiau sudarytą duomenų prognozavimo šablono modelį, buvo sudarytas tikras šablonas, kuris pavaizduotas paveikslėlyje pav. 11. Amžius nusako medžio tūrį ir aukštį, todėl jis nurodomas kaip pagrindinis laukas, pagal kurį atliekamas prognozavimas. Laukai, kuriuos reikia prognozuoti yra tūris, aukštis, ir amžius, todėl jie pažymėti *Predict*. Visi kiti laukai pažymimi kaip įėjimai.



pav. 11. Realus duomenų prognozavimo šablonas

4.3.Prognozavimo šablono veikimas

Šame skyrelyje pateikiami sudarytas prognozavimo šablonas testavimo rezultatai. Tai yra labai svarbu, nes norint pritaikyti šabloną problemai išspręsti reikia tiksliai žinoti kokius rezultatus jis gali parodyti, kokios jo paklaidos ir patikimumas.

Tam, kad patikrinti prognozavimo šablono veikimą duomenimis buvo užpildyta duomenų saugyklą. Saugyklos užpildymui naudojamas duomenų generatorius. Duomenų generatorių sudaro SQL procedūrą, kuri generuoja ir daugina įrašus duomenų saugykloje. Pirmųjų metų duomenys sukuriami metai generuojant atsitiktinius skaičius. Sekančių metų duomenis sukuriami pirmų metų duomenų pagrindu, suteikiant augimo procentinius koeficientus. Duomenų saugyklą užpildome duomenimis už 18 metų laikotarpį (nuo 1990 m. iki 2007 m.). Remiantis šitais duomenimis buvo padaryta prognozė penkeriems metams į priekį. Buvo sudaryti du eksperimentai su skirtingas tendencijas turinčiais duomenimis. Pirmame eksperimente buvo patikrinta kokius rezultatus parodo prognozavimo šablonas, kai skaičiai turi tendenciją didėti. Antrame eksperimente buvo patikrinta kokius rezultatus parodo prognozavimo šablonas, kai skaičiai turi tendenciją mažėti.

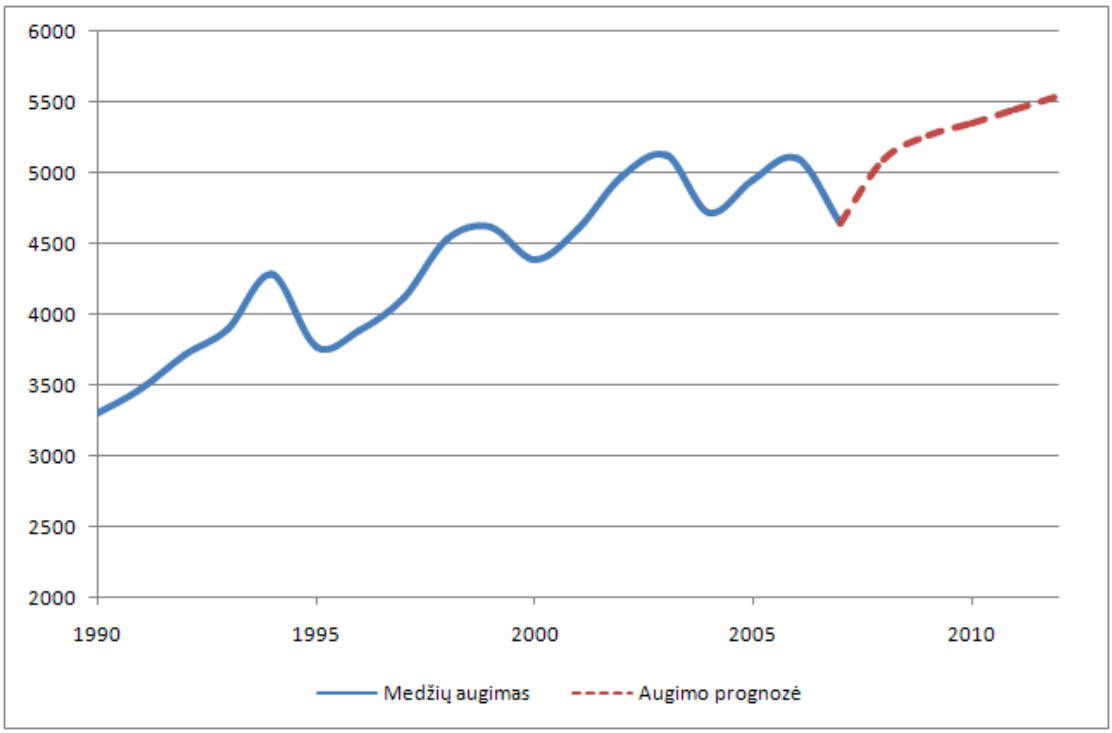
4.3.1. Duomenų prognozės rezultatai su didėjimo tendencija

Šitam eksperimentui buvo sugeneruoti duomenys su augimo koeficientais, kurie pateikti lentelėje 3.

lentelė 3. Pirmo eksperimento augimo duomenų koeficientai

Metai	Koeficientas, %	Metai	Koeficientas, %
1990	-	1999	102
1991	105	2000	95
1992	107	2001	105
1993	105	2002	108
1994	110	2003	103
1995	88	2004	92
1996	103	2005	105
1997	106	2006	103
1998	110	2007	91

Atlikus kelis šablono derinimo žingsnius buvo gauta testuojamų duomenų prognozės diagramą, kuri pavaizduota paveikslėlyje pav. 12. Tai galima interpretuoti pavyzdžiui kaip sklype augančių medžių bendro tūrio pasikeitimai laikui bėgant. Tūrio didėjimas gali reikšti natūralų medžių augimą, o tūrio mažėjimas – kirtimus. Be to, remiantis suteiktais augimo koeficientais, darome prielaidą, kad kirtimų atliekama nedaug, leidžiant didėti bendram tūriui. Laikotarpyje nuo 1990 metų iki 2007 metų pavaizduoti duomenų saugykloje esantys duomenys. Laikotarpyje nuo 2007 metų iki 2012 metų pavaizduoti šablono prognozės rezultatai. Matome, kad pagal duomenų saugykloje esančius duomenis, penkeriems metams į priekį prognozuojamas tolygus augimas su panašia tendencija.



pav. 12. Šablono prognozės rezultatai

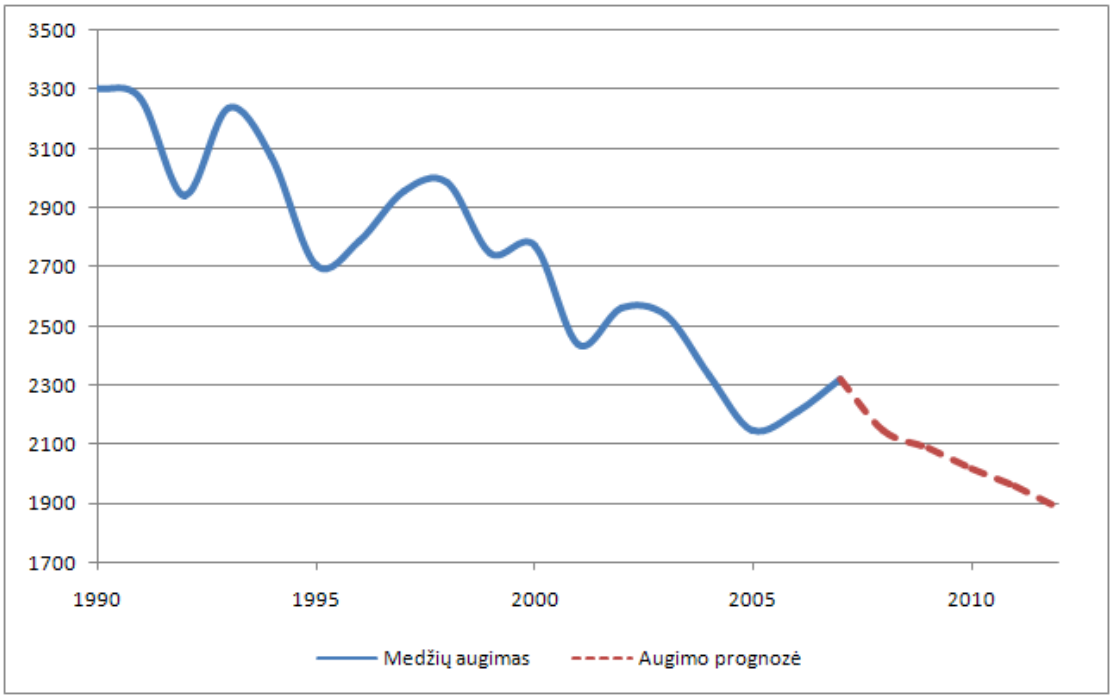
4.3.2. Duomenų prognozės rezultatai su mažėjimo tendencija

Šitam eksperimentui buvo sugeneruoti duomenys su augimo koeficientais, kurie pateikti lentelėje 4.

lentelė 4. Pirmo eksperimento augimo duomenų koeficientai

Metai	Koeficientas, %	Metai	Koeficientas, %
1990	-	1999	92
1991	99	2000	101
1992	90	2001	88
1993	110	2002	105
1994	95	2003	99
1995	88	2004	92
1996	103	2005	92
1997	106	2006	103
1998	103	2007	105

Taip pat kaip ir pirmame eksperimente, po kelių šablono derinimo žingsnius buvo gauta testuojamų duomenų prognozės diagramą, kuri pavaizduota paveikslėlyje pav. 13. Remiantis suteiktais augimo koeficientais, darome prielaidą, kad kirtimų atliekama pakankamai daug, mažinant bendram tūriui. Laikotarpyje nuo 1990 metų iki 2007 metų pavaizduoti duomenų saugykloje esantys duomenys. Laikotarpyje nuo 2007 metų iki 2012 metų pavaizduoti šablono prognozės rezultatai. Matome, kad pagal duomenų saugykloje esančius duomenis, penkeriems metams į priekį prognozuojamas tolygus mažėjimas su panašia tendencija.



pav. 13. Šablono prognozės rezultatai

4.4.Prognozavimo šablono veikimo paklaidų įvertinimas

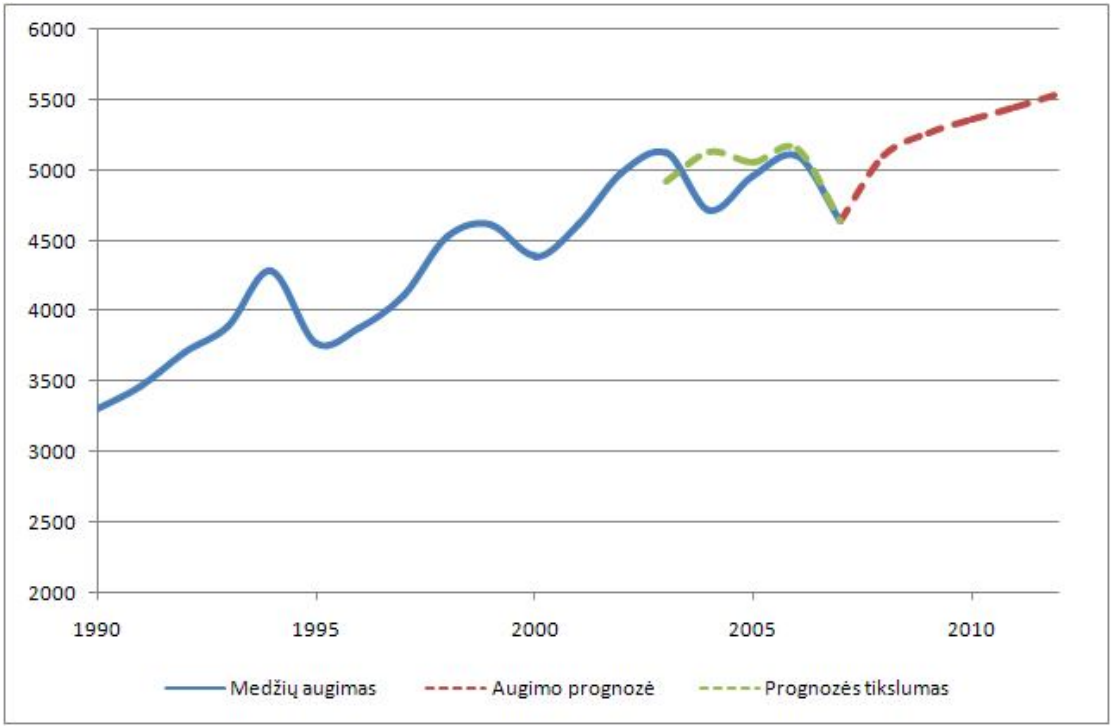
Šame skyrelyje pateikiama atliktų eksperimentų paklaidų ir jų patikimumo ribų analizė.

Pritaikant bet kokius algoritmus savo probleminei sričiai labai svarbu įvertinti algoritmo paklaidas. Algoritmo tikslumas yra vienas iš svarbiausių faktorių, pagal kuri galima daryti išvadas apie jo tinkamumą tam tikrai problemai spręsti. Geriausias būdas patikrinti sukurto prognozavimo šablono paklaidas yra atlikti atvirkštinę duomenų prognozę, t.y. prognozę tam laikotarpiui, už kurį yra realių duomenų. Darant atvirkštinę prognozė labai svarbu, kad teisingai nustatyti šablono parametrus *HISTORIC_MODEL_COUNT* (sudaromų istorinių modelių kiekis) ir

HISTORICAL_MODEL_GAP (laiko tarpas tarp istorinių modelių), nuo kurių priklauso galimas atvirkštinės prognozės laikotarpis.

4.4.1. Duomenų prognozės su didėjimo tendencija paklaidų įvertinimas

Pirmam eksperimentui atlikta atvirkštinė prognozė pavaizduota paveiksle 14. Atvirkštinė prognozė buvo atlikta laikotarpiui nuo 2007 metų iki 2003 metų. Lentelėje 5 pateiktos realių duomenų ir atvirkštinės prognozės diskrečios laikotarpio reikšmės.



pav. 14. Šablono derinimo rezultatai

Atvirkštinės prognozės paklaidas paskaičiuojamas pagal tokią formulę:

$\omega = \left|1 - \frac{r}{p}\right| \cdot 100$, kur r – reali reikšmė, p – prognozės reikšmė.

lentelė 5. Diskrečių reikšmių palyginimas

Metai	Realios reikšmės	Suprognozuotos reikšmės	Paklaida, %
2006	5101	5159	1,12
2005	4953	5059	2,09
2004	4717	5132	8,09
2003	5127	4923	4,14

Toliau reikia paskaičiuoti gautų paklaidų vidurkį ir rasti bendrą paklaidą suprognozuotam laikotarpiui. Bendrą paklaidą paskaičiuojama pagal tokią formulę:

$\omega = \frac{\sum \omega_i}{N}$, kur ω_i – yra kiekvienos diskrečios reikšmės paklaida, N – laikotarpio ilgis. Taigi bendra pirmo eksperimento paklaida ω =3,86 %.

Toliau reikia paskaičiuoti paklaidų 95% patikimumo intervalą. Tai galima padaryti naudojant tokią formulę:

$\overline{x} \mp z\frac{\sigma}{\sqrt{n}}$, kur $\overline{x}$ yra paklaidų vidurkis, σ – standartinis nuokrypis, n – imties dydis, z – reikšmė, atitinkanti norimą patikimumo lygį (paimsime z = 1,96, kas atitinka 95 % patikimumo lygį).

Pasinaudojus formulę buvo paskaičiuotas toks patikimumo intervalas:

$$3{,}86 - 1{,}96\frac{2{,}67}{\sqrt{4}} \leq \mu \leq 3{,}86 + 1{,}96\frac{2{,}67}{\sqrt{4}}$$

Taigi paklaidos patikimumo intervalas yra:

$$1{,}24 \leq \mu \leq 6{,}48$$

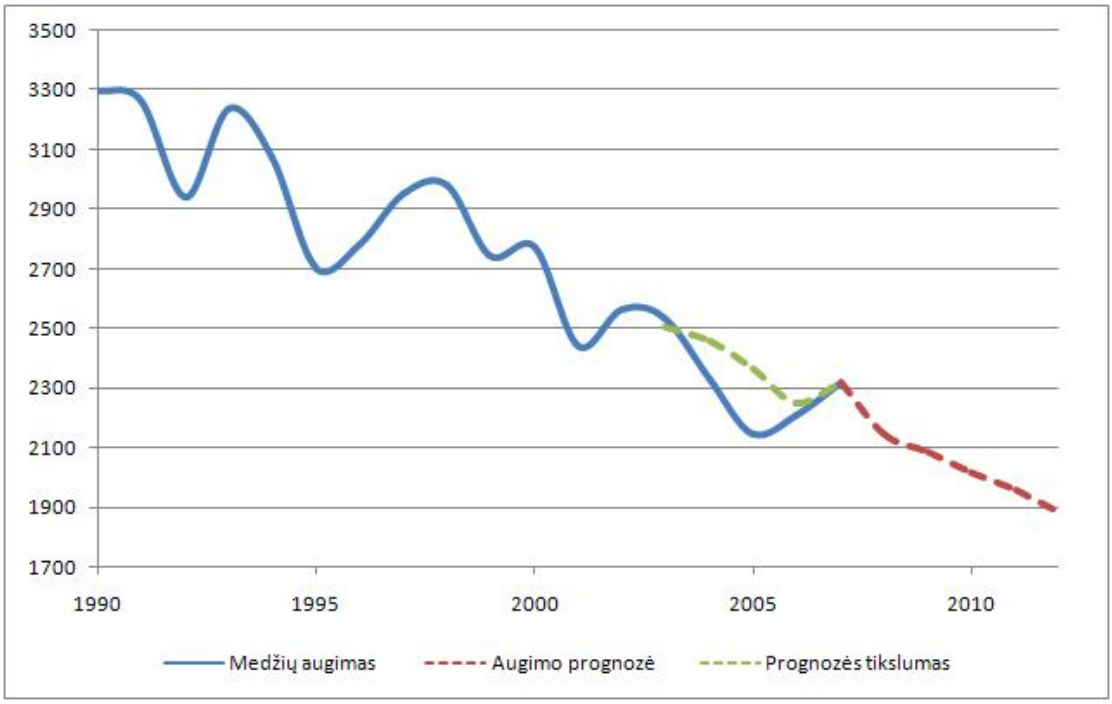
Matome, kad eksperimento rezultatų paklaida, kurios reikšmė yra 3,86%, patenka į patikimumo intervalą.

4.4.2. Duomenų prognozės su mažėjimo tendencija paklaidų įvertinimas

Antram eksperimentui atlikta atvirkštinė prognozė pavaizduota paveiksle 15. Atvirkštinė prognozė atlikta laikotarpiui nuo 2007 metų iki 2003 metų. Lentelėje 6 pateiktos realių duomenų ir atvirkštinės prognozės diskrečios laikotarpio reikšmės ir paklaidos.

Lentelė 6. Diskrečių reikšmių palyginimas

Metai	Realios reikšmės	Suprognozuotos reikšmės	Paklaida, %
2006	2211	2249	1,69
2005	2147	2366	9,25
2004	2333	2457	5,05
2003	2536	2504	1,28



pav. 15. Šablono derinimo rezultatai

Antro eksperimento atvirkštinės prognozės diskrečių reikšmių paklaidas skaičiuojamos taip pat kaip ir pirmame eksperimente.

Paskaičiavus visų šio eksperimento diskrečių reikšmių paklaidų vidurkį, buvo gauta antro eksperimento bendra paklaida ω =4,32 %.

Toliau galia paskaičiuojamas paklaidų 95% patikimumo intervalas. Tai atliekama naudojant tokią formulę:

$\overline{x} \mp z\frac{\sigma}{\sqrt{n}}$, kur $\overline{x}$ yra paklaidų vidurkis, σ – standartinis nuokrypis, n – imties dydis, z – reikšmė, atitinkanti norimą patikimumo lygį (čia geriausiai paimti z = 1,96, kas atitinka 95 % patikimumo lygį).

Pasinaudojus formulę buvo gautas toks patikimumo intervalas:

$$4{,}32 - 1{,}96\frac{3{,}2}{\sqrt{4}} \leq \mu \leq 4{,}32 + 1{,}96\frac{3{,}2}{\sqrt{4}}$$

Taigi paklaidos patikimumo intervalas yra:

$$1{,}18 \leq \mu \leq 7{,}45$$

Matome, kad eksperimento rezultatų paklaida, kurios reikšmė yra 4,32%, patenka į patikimumo intervalą.

4.4.3. Eksperimentų paklaidų apibendrinimas

Atlikus du eksperimentus paaiškėjo, kad prognozavimo šablonas gali nustatyti duomenų tendencijas ir daryti prognozes. Abiejų eksperimentų metu gautos labai artimos paklaidos, todėl galia padaryti išvadą, kad prognozavimo šablonas veikia korektiškai. Reikia pažymėti, kad šablono prognozės paklaidos priklauso ir nuo duomenų kiekio ir nuo sudarytų istorinių modelių kiekio, bei kokybės. Šiam eksperimentui atlikti buvo naudojama duomenų saugykla, kurioje buvo apie 60000 įrašų.

Kitas klausymas yra ar toks šablonas yra tinkamas problemai išspręsti. Iš abiejų eksperimentų buvo gauta vidutinė 4% paklaida ir galima teigti kad toks rezultatas yra patenkinamas. Medžių prognozavimui tokia paklaida yra priimtina, nes ir faktinė informacija gali būti netiksli pavyzdžiui lyginant su pardavimų informaciją.

4.5.Duomenų saugyklos užpildymas trūkstamais duomenimis

Šiame skyrelyje aprašomas duomenų saugyklos užpildymo procese dalyvaujantys elementai. Pagal 3.3 skyriuje aprašyta duomenų saugyklos užpildymo proceso modelį buvo sudarytos užklausos, kurios realizuoja viso proceso grandinę.

4.5.1. Filtruojanti MDX užklausa

Filtruojančio MDX užklausos paskirtis yra paruošti prognozavimui tinkamą duomenų sluoksnį. Duomenų sluoksnį turi sudaryti nurodyto laikotarpio vieno sklypo faktiniai duomenis. Ši užklausa gali turėti pastovią formą, o jos filtruojančios dimensijų reikšmės turi būti keičiamos priklausomai nuo analizei reikalingo duomenų sluoksnio.

Filtruojančios MDX užklausos pavyzdys pateiktas žemiau. Tai yra pakankamai paprasta MDX užklausa, kurioje filtruojami duomenis pagal tokias dimensijas:

1. Laiko dimensija – tai dimensija, kuri turi filtruoti duomenis pagal reikiamą laikotarpį. Čia turi būti išvardinami metai, pagal kuriuos bus atliekama analizė.
2. Medžio tipo dimensija – tai dimensija, kuri filtruoja duomenis pagal medžio tipą. Čia visada turi būti viena filtruojanti reikšmė.

3. Miško tipo dimensija – tai dimensija, kuri filtruoja duomenis pagal miško tipą.
4. Inventorizacijos metų dimensija – tai dimensija, kuri filtruoja duomenis pagal inventorizacijos metus. Čia visada turi būti viena reikšmė.
5. Sklypo dimensija – tai dimensija, kuri filtruoja duomenis pagal sklypą. Čia visada turi būti viena reikšmė.
6. Vietovės dimensija – tai dimensija, kuri filtruoja duomenis pagal vietovę, kurioje randasi sklypas. Čia visada turi būti viena reikšmė.

```
SELECT
{
                [TIME DIM].[Time Year].&[2000],
                [TIME DIM].[Time Year].&[2001], ...
} ON COLUMNS
FROM
(
        SELECT
                {[TREE KIND DIM].[Tk Name].&[Ąžuolas]}
                ON COLUMNS
        FROM
        (
                SELECT
                        {[FOREST TYPE DIM].[Ft Name].&[Valstybinis miškas]}
                        ON COLUMNS
                FROM
                (
                        SELECT
                                {[INVENTORY DIM].[INVENTORY DIM].&[1]}
                                ON COLUMNS
                        FROM
                        (
                                SELECT
                                        {[QUARTER DIM].[q Name].&[KV_1]}
                                        ON COLUMNS
                                FROM
                                (
                                        SELECT
                                                {[LOCATION DIM].[Loc Name].&[Biržų k.]}
                                                ON COLUMNS
                                        FROM [Cutting DW Cube]
                                )
                        )
                )
        )
)
```

Visos dimensijos užklausos šablone gali būti naudojamos bet kuria tvarka. Kai užklausos šablonas užpildomas, gaunamas filtras, kurį galima naudoti sekančiame etape – šablono užpildymo užklausos sudaryme.

4.5.2. Šablono užpildymo užklausa

Šablono užpildymo užklausos paskirtis yra užpildyti šabloną analizei reikiamais duomenimis. Žemiau pateikiamas tokios užklausos pavyzdys. Ši užklausa sudaroma iš MDX ir DMX užklausų derinio. DMX užklausa turi įrašyti į

prognozavimo šabloną paduodamus duomenis. Duomenis paduodami su MDX užklausos pagalba, kuri suteikia duomenims su šablono įėjimais suderinamą formą. Į šią MDX užklausa integruojama ir filtruojanti MDX užklausa. Tai atliekama MDXFilter bloko viduje.

```
INSERT INTO MINING STRUCTURE [SINGLECUTTING]
(SKIP, [TIME DIM], [Age], [Diameter], [Height])
OPENQUERY
(
        [Cutting DW],
        'SELECT
                DATAID ([Cutting DW Cube].[PARCEL FACT].[$TIME DIM.TIME DIM])
                AS [SKIP TIME],
                KEY ([Cutting DW Cube].[PARCEL FACT].[$TIME DIM.TIME DIM], 0)
                AS [TIME DIM],
                AGGREGATE ([Cutting DW Cube].[PARCEL FACT].[Age])
                AS [Age],
                AGGREGATE ([Cutting DW Cube].[PARCEL FACT].[Diameter])
                AS [Diameter],
                AGGREGATE ([Cutting DW Cube].[PARCEL FACT].[Height])
                AS [Height]
        FROM
                [Cutting DW Cube].[PARCEL FACT]
        WHERE
        (
                MDXFilter ("Filtruojanti MDX Užklausa")
        )
        GROUP BY
                [SKIP TIME], [TIME DIM]
        ORDER BY
                DATAID ([Cutting DW Cube].[PARCEL FACT].[$TIME DIM.TIME DIM]) ASC'
)
```

Tai yra duomenų prognozavimo šablono užpildymo užklausos šablonas. Ši užklausa gali turėti pastovią struktūrą, bet kiekviename žingsnyje turi būti keičiama filtruojanti MDX užklausa.

4.5.3. Prognozavimo šablono užklausa

Prognozavimo šablono užklausos paskirtis yra gauti prognozuojamus duomenis. Žemiau pateikiamas tokios DMX užklausos pavyzdys. Ši užklausa iš duomenų prognozavimo šablono *[SINGLECUTTING]* grąžina penkeriems metams suprognozuotas reikšmes. Užklausos rezultatai talpinami į tarpinę lentelę iš kurios jie bus paimti sekančio žingsnio vykdymo metu.

```
INSERT INTO [TEMP RESULTS TABLE]
SELECT FLATTENED
  PredictTimeSeries([SINGLECUTTING].[Age], 5),
  PredictTimeSeries([SINGLECUTTIN1].[Diameter], 5),
  PredictTimeSeries([SINGLECUTTIN1].[Height], 5),
FROM
  [SINGLECUTTING]
```

Ši užklausa gali turėti pastovią konstrukcija, t.y. galima naudoti pavyzdį kaip šabloną. Prognozavimo laikotarpis gali būti nurodomas parametru, nes įvairiais atvejais gali prireikti skirtingo laikotarpio duomenų.

4.5.4. Transformavimo užklausa

Transformavimo užklausos paskirtis yra suteikti suprognozuotiems duomenims saugyklos faktų lentelės formatą ir papildyti jais duomenų saugyklą. Kadangi šiame žingsnyje duomenis turi detalumą tik laiko atžvilgiu, tai juos reikia transformuoti, pridedant prie jų kitų dimensijų reikšmes. Žemiau pateikiamas tokios SQL užklausos pavyzdys. Kadangi prognozavimo rezultatai saugomi į paprastą lentelę *[TEMP RESULT TABLE]*, tai ir filtruojančiai MDX užklausos šablonui paduodamų parametrų reikšmes galima įrašyti į lentelę *[TEMP DIMENSION VALUES TABLE]*. Atlikus elementarų duomenų išrinkimą iš dviejų lentelių galime gauti reikiamo formato duomenis.

```
INSERT INTO [Cutting DW].[Parcel Fact]
(tk_code, ft_code, q_code, loc_code, inv_code, time_code, age, height, diameter)
SELECT
       p.[TREE TYPE DIM],
       p.[FOREST TYPE DIM],
       p.[QUARTER DIM],
       p.[LOCATION DIM],
       p.[INVENTORY DIM],
       r.[TIME DIM],
       r.[AGE],
       r.[DIAMETER],
       r.[HEIGHT]
 FROM
       [TEMP RESULTS TABLE] r,
       [TEMP DIMENSION VALUES TABLE] p
```

Užklausos grąžinami duomenys gali būti įkeliami į duomenų saugyklą. Čia galima pridėti *insert* konstrukciją, kuri transformuotus duomenis gali įkelti į saugyklos faktų lentelę. Tokia užklausa gali turėti pastovią konstrukciją.

4.6.Prognozės rezultatų pateikimas

Darbui su duomenų saugyklą buvo sudarytas kubas, kurio struktūra atitinka paveiksle 9 pavaizduotą saugyklos schemą. Duomenų analizei ir peržiūrai buvo sudaryta ataskaitą, kuri pavaizduota paveiksle 16. Ataskaitai naudojamas *Microsoft Excel 2007* įrankis, kuris yra labai patogus, paprastas ir tuo pačių galingas įrankis OLAP ataskaitoms kurti. Ataskaitoje pavaizduotas kiekvienais metais bendras medžių tūris pagal apskritis su vietovių detalizacija.

Inv Name	All				
Ft Name	All				
Fe Name	All				
For Name	All				
q Name	All				
Diameter	Column Labels				
Row Labels	1995	1996	1997	1998	Grand Total
⊟Alytaus apskritis	5338,72	6228,56	7118,36	8008,16	26693,8
⊞Druskininkų m.	2885,35	3366,26	3847,15	4328,03	14426,79
⊞Grūto k.	2453,37	2862,3	3271,21	3680,13	12267,01
⊟Kauno apskritis	5007,57	5842,19	6676,81	7511,43	25038
⊞Jonavos m.	2431,8	2837,11	3242,41	3647,71	12159,03
⊞Upninkų k.	2575,77	3005,08	3434,4	3863,72	12878,97
⊟Marijampolės apskritis	2739,54	3196,13	3652,75	4109,36	13697,78
⊞Vilkaviškio m.	2739,54	3196,13	3652,75	4109,36	13697,78
⊟Šiaulių apskritis	7119,02	8305,53	9492,04	10678,55	35595,14
⊞Joniškio m.	2575,79	3005,09	3434,39	3863,68	12878,95
⊞Šiaulių m.	2142,04	2499,04	2856,04	3213,07	10710,19
⊞Žagarės m.	2401,19	2801,4	3201,61	3601,8	12006
Grand Total	20204,85	23572,41	26939,96	30307,5	101024,72

pav. 16. Prognozės rezultatai

4.7.Eksperimento išvados

Kaip jau buvo kalbėta anksčiau, tikrojo medžio augimas yra tolydus reiškinys, ir jeigu labai stiprių išorinių poveikiu, medis ilgą laiką tolygiai auga. Todėl tokios informacijos duomenų saugyklos papildymas neturi būti labai dažnas, bet kartais gali praversti apytiksli informaciją apie tam tikrą sritį Duomenų prognozavimas gali nustatyti tam tikras tendencijas, kurios gali būti labai artimos realybei.

Eksperimento metu buvo atliktas prognozavimas penkiems metams į priekį ir keturiems metams atgal ir buvo nustatyta, kad prognozavimo šablono parodyti rezultatai gali būti labai artimi realiai situacijai.

Visą duomenų saugyklos užpildymo procesą sudaro keturi užklausų šablonai. Šablonų realizavimui prireikė trijų užklausų kalbų SQL, MDX ir DMX. Sudėtingiausios yra pirmų dviejų etapų duomenų išrinkimui skirtos MDX užklausos. Kitų etapų užklausos gali būti žymiai paprastesnės.

5. Išvados

1. Šiuo metu rinkoje yra labai daug veiklos intelekto priemonių. Rinkai sparčiai besikeičiant, sparčiai tobulinami ir įrankiai. Tokių įrankių poreikį diktuoja informacinių sistemų sudėtingumo augimas ir labai greitas duomenų kiekių didėjimas. Netgi populiariausių veiklos intelekto įrankių išvystymo lygis, bei siūlomo funkcionalumo apimtis yra labai skirtingi. Microsoft Analysis Services yra vienas iš geriausių įrankių, turintis praktiškai visas veiklos intelekto naujoves. Tai yra vienas ir pagrindinių kriterijų, kodėl jis buvo pasirinktas eksperimentams atlikti.
2. Kaip parodė duomenų išrinkimo kalbų analizė, SQL ir MDX turi daug skirtumų. Nors SQL kalba ir gali būti naudojama darbui su duomenų saugyklomis, bet ji tam nėra gerai pritaikyta, todėl užklausos gaunasi sudėtingesnės, o duomenų išrinkimo greitis yra neoptimalus. MDX kalba yra pritaikyta tik darbui su daugiamačiais duomenimis, todėl ji pranoksta SQL kalbą šioje srityje ir pagal veikimo greitį ir pagal pateikiamų rezultatų patogumą ir pagal kodo apimtį. Reikia pažymėti kad MDX užklausų sintaksė šiek tiek sudėtingesnė nei SQL kalbos. DMX kalbos specifika skiriasi nuo dviejų paminėtų kalbų, nes ji skirta ne tiesioginei duomenų analizei, o analizei, išgaunant duomenis papildančią, apibendrinančią arba juos klasifikuojančią informaciją. Be to, DMX kalba gerai integruojasi tiek su SQL, tiek su MDX kalbomis ir netgi negali apsieiti be jų, nes su šitų kalbu pagalba DMX gauna analizuojamus duomenis.
3. Išanalizuotas duomenų prognozavimo šablonas, naudojantis Microsoft Time Series algoritmą. Tai yra statistinis algoritmas, veikiantis pagal autoregresijos ir sprendimų medžio algoritmų principus. Jo tikslumas, kaip ir kiekvieno statistinio algoritmo, priklauso nuo duomenų kokybės, todėl naudoti duomenų generatorius yra pakankamai rizikinga. Nepaisant to, netgi naudojant duomenų generatorių buvo pasiekta tik 4% paklaida, kas yra tikrai geras pasiekimas. Eksperimentui sudarytas prognozavimo šablonas yra gana universalus ir jį galima būtų pritaikyti kelioms problemoms, kurios turi panašius išėjimus. Bet šiuo metu toks šablonas dar negali būti universalus programinės įrangos atžvilgiu. PMML klaba, kurios tikslas padaryti duomenų gavybos šablonus universaliais platformos prasme, dar nepalaiko duomenų prognozavimo algoritmų.

4. Iškeltai problemai spręsti sudarytas duomenų prognozavimo šablono testavimas parodė labai gerus rezultatus. Kaip parodė probleminės srities analizė, problemai išspręsti nebūtina turėti labai tikslų prognozavimo šabloną, todėl galima teigti, kad šablonas, turintis 4% paklaidą, yra tinkamas.
5. Eksperimentas parodė, kad duomenų gavybos funkcijas galima naudoti duomenų saugykloje trūkstamų duomenų sukūrimui. Kadangi prognozavimo šablonui negalima suteikti duomenų saugyklos faktų lentelės formatą, uždavinys tampa sudėtingesnis. Dėl to, kad šablono įėjime gali būti tik laiko dimensija ir faktiniai skaičiai, visas papildymo procesas pavirsta duomenų filtravimo, perrašymo ir transformavimo veiksmų seka. Toks sprendimas gali būti neoptimalus laiko prasme, nes prognozavimas atliekamas cikliškai kiekvienai vieno tipo duomenų sluoksnio aibei. Jeigu šablonui galima būtų suteikti duomenų saugyklos faktų lentelės formatą, nurodant pagrindinį laiko raktinį lauką, dimensijų raktinius laikus ir faktinių skaičių laukus, tai papildymo proceso grandinė galėtų būti trupesnė, o pats procesas būtų optimalesnis.
6. Visam duomenų saugyklos papildymo procesui realizuoti prireikė trijų užklausų kalbų: SQL, MDX ir DMX. MDX užklausos buvo panaudotos analizei reikiamų duomenų kubo sluoksnių gavimui. DMX užklausos buvo panaudotos prognozavimo šablono papildymui ir rezultatų gavimui. SQL užklausos buvo panaudotos plokščių duomenų transformavimui ir jų įkėlimui į duomenų saugyklą. Reikia pažymėti, kad prognozavimo šablono užkrovimui buvo panaudotas MDX ir DMX užklausų derinys. Tai parodo kad eksperimentui naudojamas įrankis pasižymi geru skirtingu kalbų suderinamumu. Visos sudarytos užklausos nebuvo labai sudėtingos, nes užduočiai atlikti neprireikė labai sudėtingų manipuliavimų duomenimis.

6. Literatūra

[1] George Spofford, Sivakumar Harinath, Christopher Webb, Dylan Hai Huang, Francesco Civardi. MDX Solutions Second Edition. Wiley Pubishing inc.

[2] Ralph Kimball, Margy Ross. The Data Warehouse Toolkit. The Complete Guide to dimensional Modeling. John Wiley & Sons, Inc.

[3] Eric Thomson. OLAP Solutions. Building Multidimensional Information Systems. Second edition. Published by John Wiley & Sons, Inc. 2002.

[4] Tony Bain, Mike Benkovich, Robin Dewson, Sam Ferguson, Christopher Graves, Terrence J. Joubert, Denny Lee, Mark Scott, Robert Skoglund, Paul Turley, Sakhr Youness. Professional SQL Server 2000 Data Warehousing with Analysis Services. *Wrox Press Ltd. 2001.*

[5] Sivakumar Harinath Stephen R. Quinn. Professional SQL Server Analysis Services 2005 with MDX. Wrox Press 2006.

[6] W. H. Inmon. Building the Data Warehouse Third Edition. John Wiley & Sons, Inc.

[7] MSDN Library. Comparison of SQL and MDX. Prieiga per internetą: http://msdn2.microsoft.com/en-US/library/aa216779(sql.80).aspx

[8] MSDN Library. Introduction to the Unified Dimensional Model (UDM). Prieiga per internetą: http://msdn2.microsoft.com/en-us/library/ms345143(SQL.90).aspx

[9] by Lark "Group, Inc.". Why you should add MDX to your SQL Server toolkit. Jul 21, 2003. Prieiga per internetą: http://articles.techrepublic.com.com/5100-22_11-5054229.html

[10] Stéphane Faroult with Peter Robson. The Art of SQL. O'Reilly Media, Inc 2006.

[11] ZhaoHui Tang, Jamie MacLenan. Data Mining With SQL Server 2005. Wiley Publishing Inc.

[12] C. Meek, D. M. Chickering, D. Heckerman. Autoregressive Tree Models For Time-Series Analysis, 2002.

[13] P. Smith, G. Renganathan, G. Das, K. Lin, H. Manila. Rule Discovery From Time Series. Prieiga per internetą: *http://citeseer.ist.psu.edu/das98rule.html*

7. Terminų ir santrumpų žodynas

Šiame skyriuje atskirai pateikiami santrumpų ir terminų žodynai.

7.1.Santrumpos

lentelė 6. Santrumpos

Santrumpa	Apibūdinimas
SQL	Structured Query Language
MDX	Multidimensional Expressions
DMX	Data Mining Extensions
BI	Business intelligence
OLAP	Online Analytical Processing
UDM	Unified Dimensional Model
ROLAP	Relational Online Analytical Processing
MOLAP	Multidimensional Online Analytical Processing
HOLAP	Hybrid Online Analytical Processing
PMML	Predictive Model Markup Language
ETL	Extract Transform Load

7.2.Terminai

lentelė 7. Terminai

Terminas	Terminas anglų kalba (English)	Apibūdinimas
Veiklos Intelektas	Buisiness Intelligence	Technologijų ir programinės įrangos aibė, kurios skirtos duomenų surinkimui, integracijai, atvaizdavimui ir analizei.
Duomenų gavyba	Data Mining	Duomenų analizės procesas, kai bandoma atrasti iš pirmo žvilgsnio nepastebimas duomenų savybes.

8. Priedai

8.1.Straipsnis

Daugiamačių duomenų analizės galimybių išplėtimas taikant šablonus

Aleksandr Murašov

Kauno Technologijos Universitetas, Studentų g. 50, LT – 51368 Kaunas

Straipsnyje nagrinėjamos galimybės padidinti duomenų analizės efektyvumą taikant daugiamačių duomenų užklausų bei analizės kalbas MDX ir DMX. Šios kalbos yra mažai žinomos ir jų galimybės nėra pilnai išnaudojamos. Pavyzdžiui, duomenų saugyklos užklausas galima papildyti duomenų gavybos metodų funkcijomis, kurios iš karto pateikia reikiamus rezultatus. Šiame straipsnyje nagrinėjamos DMX šablonų taikymo galimybės prognozuojant trūkstamas duomenų reikšmes.

1 Įvadas

Didelė dalis įmonių kuria arba yra numatę artimiausioje ateityje kurti duomenų saugyklas ir naudoti duomenų analizės bei verslo intelekto priemones. Šioje srityje teorija yra gana jauna ir nėra daugiamačių užklausų kalbos standarto, analogiško SQL (angl. *Structured Query Language*). SQL kalba galima kurti daugiamačių duomenų užklausas, tačiau pačios užklausos ir jų rašymas nėra efektyvus. Patyrę duomenų saugyklų kūrėjai naudoja daugiamačių duomenų reiškinių MDX (angl. *Multidimensional Expressions*) ir duomenų gavybos išplėtimų DMX (angl. *Data Mining Extensions*) užklausų kalbas, kurios yra efektyvesnės daugiamačių duomenų analizei, nei SQL, ir turi plėtimo galimybes. Tačiau šios kalbos kol kas nėra pripažintos standartu ir jų potencialios galimybės nėra iki galo išnaudojamos. Šio straipsnio tikslas – parodyti vieną iš galimybių, kaip galima paspartinti duomenų analizei ir veiklos intelektui skirtų sprendimų kūrimą, pritaikant DMX šablonus.

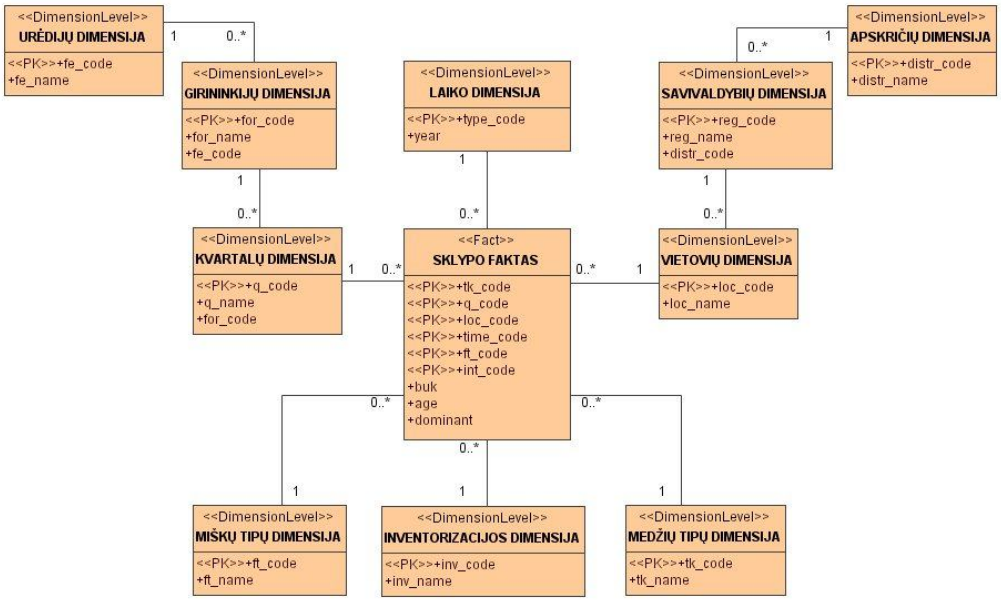
Straipsnyje aprašytas DMX šablonų taikymas realiai problemai spręsti. Pavyzdžiui, Lietuvos miškų informacinėje sistemoje faktiniai duomenys apie medžių aukštį, storį, brandumą ir tinkamumą kirtimui įvairiose vietovėse renkami tik kas penkeri metai. Kadangi medžiai auga gana tolygiai, užpildyti trūkstamas reikšmes galima panaudojant prognozavimo algoritmus. Šią užduoti efektyviai galima atlikti, sukuriant miškų istorinės informacijos saugyklą ir pritaikant DMX.

2 Duomenų saugyklų kūrimas

Tipinė duomenų saugykla skiriasi nuo reliacinės duomenų bazės. Pirmiausiai, paprastos duomenų bazės skirtos tam, kad padėtų vartotojams atlikti kasdieninį darbą, o duomenų saugyklos skirtos sprendimų priėmimui. Paprastos duomenų bazės duomenis keičiasi labai greitai, o duomenų saugyklos duomenis keičiasi žymiai lėčiau: duomenis dažniausiai atnaujinami pagal tvarkaraštį (kas valandą, kas savaitę, kas mėnesį), priklausomai nuo poreikio [5]. Idealiu atveju duomenų saugyklos papildymo procesas yra paprastas tam tikro laiko tarpo agreguotų duomenų įkėlimas nekeičiant saugykloje esančių duomenų. Tam tikslui naudojama ETL (angl. *Extract, Transform, Load*) technologija, kurios esmė yra paimti duomenis iš duomenų šaltinio, transformuoti juos pagal duomenų saugyklos struktūrą ir įrašyti juos į duomenų saugyklą [1]. Duomenų šaltiniu dažniausiai būna reliacinės duomenų bazės, bet gali būti ir struktūrizuoti dokumentai.

Daugiamatę duomenų saugyklą sudaro faktų lentelė ir dimensijų lentelės. Faktų lentelė yra pagrindinė duomenų saugykloje. Joje saugoma informacija apie objektus arba įvykius, kurie bus analizuojami. Dažniausiai naudojami faktai susiję su transakcijomis, momentiniais įvykiais, objekto būsena ir t.t. Dimensijų lentelės detalizuoja analizuojamus faktus pagal įvairius pjūvius. Išskiriami du daugiamačių saugyklų struktūros tipai: žvaigždė ir snaigė. Esminė žvaigždės schemų savybė yra dimensijų paprastumas. Tokios schemos dimensijos neturi hierarchijų. Snaigės tipo duomenų saugyklos gali būti sudėtingesnės, nes dimensijos gali turėti hierarchijas [2]. Miškų informacijos saugyklos schema pateikiama 1 paveiksle.

Tipinės daugiamačių saugyklų struktūros nėra normalizuotos, o tai leidžia pasiekti labai didelį duomenų išrinkimo greitį bei labai aukštą duomenų analizės lankstumo lygį. Duomenų išrinkimui iš daugiamačių duomenų struktūrų naudojama speciali kalba MDX. Ši užklausų kalba skirta darbui su daugiamatėmis struktūromis. Nors MDX kalbos sintaksė gali pasirodyti labai panaši į SQL kalbos sintaksę, bet tarp šitų užklausų kalbų yra labai didelis skirtumas. SQL kalbą galima naudoti darbui su daugiamatėmis struktūromis, bet ji tam nėra labai tinkama, nes užklausų sudėtingumas būtų labai didelis, o duomenų išrinkimo laikas nebūtų optimalus. MDX kalba pritaikyta darbui tik su daugiamačiais duomenimis ir leidžia pasiekti labai didelį duomenų išrinkimo greitį [6].



1 pav. Miškų informacijos saugykla

Viena iš geriausių, populiariausių ir sparčiausiai besivystančių veiklos intelekto (angl. *Business Intelligence*) sistemų yra *Microsoft SQL Server Analysis Services*. Egzistuoja ir daug kitų veiklos intelekto sistemų, pavyzdžiui *Oracle Database for Business Intelligence and Data Warehousing*, *IBM InfoSphere Warehouse*, *SAS Business Intelligence* kartu su *SAS Analytics*. Išvardinti produktai palaiko OLAP (angl. *Online Analytical Processing*) ir duomenų gavybos (angl. *Data Mining*) technologijas, bet jų išvystymo lygis labai skiriasi. Pavyzdžiui, kiekvienas produktas turi savo duomenų gavybos algoritmus, kurių kiti produktai gali neturėti. Nėra priimtų standartų nei duomenų gavybos algoritmams, nei duomenų gavybos kalbai. Bet DMG (angl. *Data Mining Group*) grupė aktyviai dirba ties to ir vysto PMML (angl. *Predictive Mining Model Language*) kalbą, kuri XML pagalba leidžia aprašyti universalius nuo veiklos intelekto sistemos nepriklausomus duomenų gavybos šablonus. Į šią grupė įeina tokios korporacijos kaip *Microsoft*, *IBM*, *Oracle*, *SAS* ir kiti didžiausi verslo intelekto sistemų kūrėjai. Šiuo metu didžioji dalis siūlomų duomenų gavybos algoritmų yra nesuderinami su PMML klaba aprašytais šablonais, todėl šiuo metu šablono kūrimas ir naudojimas labai priklauso nuo pasirinktos sistemos.

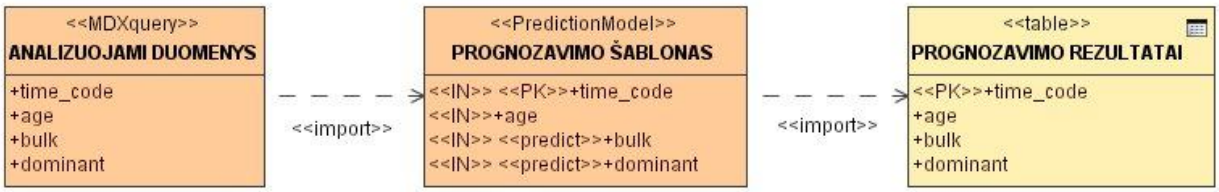
Duomenų gavyba yra naudingų duomenų apie tam tikrą veiklą gavimo metodų aibė. Ši technologija suteikia funkcijas, kurios leidžia aptikti duomenų saugykloje esančių duomenų dėsningumus, panašumus ir atlikti duomenų prognozavimą, klasifikavimą, grupavimą. Tai naudinga, nes saugomų duomenų kiekiai auga labai greitai, ir visus duomenis išnagrinėti yra labai sudėtinga, o duomenų gavybos funkcijos gali aptikti nematomas duomenų savybes, kurių analizė padės priimti teisingą veiklos strategiją.

Su veiklos intelekto funkcijomis galima išspręsti tokius uždavinius: investicijų grąžos analizė, pardavimų analizė ir prognozavimas, apgavysčių nustatymas, rizikos valdymas, klientų skaidymas, reklamos efektyvumo analizė. Visus spręstinus uždavinius pagal sprendimo būdą galima sugrupuoti į kelias grupes ir apibrėžti universalius sprendimo metodus:

- Klasifikavimas – šiš metodas priskiria tam tikrus požymius duomenų aibėms, turinčioms tam tikrus bendrus atributus. Tokius atributus dažniausiai reikia aptikti, nes jie nėra akivaizdūs. Klasifikavimas gali būti atliekamas naudojant sprendimų medžių, *Naive Bayes* algoritmus ir neuroninius tinklus [4].
- Grupavimas (*skaidymas*) – šis metodas naudojamas tada, kai pagal tam tikrus atributus reikia aptikti natūralias duomenų grupes. Visi įėjimo atributai apdorojami vienodai ir tam, kad aptikti atributų bendrus požymius, reikia atlikti labai daug duomenų apdorojimo iteracijų, kurios stabdomos tada, kai stabilizuojasi grupių ribos [4].
- Asociacijų paieška – šis metodas randa dažnai kartu naudojamų objektų aibes. Pavyzdžiui, prekės, kurios dažniausiai patenka į vieną krepšelį. Dauguma algoritmų, su kuriais realizuojamas šis metodas, skenuoja duomenis kelis kartus, be to, dažniausiai dar reikia nurodyti pasikartojimo dažnio slenkstį ir bendrų objektų aibės dydį [4].

- Regresija - šis metodas yra panašus į klasifikavimą. Pagrindinis skirtumas yra tas, kad nuspėjamas atributas yra tolydus skaičius. Regresijos tipo uždaviniams spręsti dažniausiai naudojami linijinės ir loginės regresijos metodai, bet sprendimui taip pat galima regresijos medžius ir neuroninius tinklus [4].
- Prognozavimas – šis metodas dažniausiai naudojamas pardavimų prognozėms daryti. Įėjimas dažniausiai būna laiko dimensija. Tokių uždavinių sprendimui dažniausiai naudojamas ARIMA (angl. *Autoregressive Integrated Moving Average*) metodas [4].
- Sekų analizė – šis metodas naudojamas tada, kai reikia rasti bendras savybes diskrečių reikšmių aibėje. Tai yra pakankamai nauja uždavinių aibė, kurių sprendimui galima pritaikyti Markovo grandinių metodą. Mokslininkai šiuo metu ieško naujų algoritmų, kurie būtų tinkami tokiems uždaviniams spręsti [4].
- Nukrypimų analizė - šis metodas taikomas tada, kai reikia rasti tokius objektus, kurie labai skiriasi nuo bendros objektų aibės. Šiuo metu nėra standartinių metodų nukrypimo analizei atlikti ir šitoje srityje vykdoma daug tyrimų, bet dažniausiai naudojamos sprendimų medžių grupavimo ir neuroninių tinklų algoritmų modifikacijos [4].

Pateiksime vieną iš *Data Mining* pritaikymo pavyzdžių, naudojant duomenų gavybos išplėtimų kalbą DMX. Ši kalba naudojama *Microsoft SQL Server Analysis Services* sistemoje. Viena iš perspektyviausių DMX galimybių – pritaikyti šablonus, kurie leidžia tipizuoti veiklos analizės sprendimus. Duomenų gavybos šabloną galima įsivaizduoti kaip lentelę į kurią įrašius duomenų porciją galima gauti apdorotų duomenų porciją. Duomenų gavybos šablonas įvade aprašytai miškų informacijos sistemos duomenų prognozavimo problemai spręsti pateikiamas 2 paveiksle.



2 pav. Duomenų gavybos šablono modelis

Tai yra prognozavimo šablono modelis, kuris turėtų prognozuoti medžių augimą laikui bėgant. Šablonas turi įėjimus ir išėjimus, kuriems suteikiami tam tikri požymiai. Pavyzdžiui, laikas yra atraminis laukas, pagal kuri vykdoma analizė, o tūris ir aukštis yra analizuojami ir prognozuojami laukai. Duomenis į šabloną gali būti perduodami su MDX užklausų pagalba, kurios grąžinamų duomenų formatas turi būti suderinamas su šablono įėjimais [3].

DMX kalba leidžia sukurti šablonus, kurie duomenų analizei naudoja tokius algoritmus: sprendimo medžių algoritmus; grupavimo algoritmus; *Naive Bayes* algoritmus; asociacijų paieškos algoritmus; sekų grupavimo algoritmus; laiko eilučių prognozavimo algoritmus; neuroninių tinklų algoritmus; strateginės regresijos algoritmus; tiesinės regresijos algoritmus. Visų išvardintų šablonų nenagrinėsime, nes jų koncepcija yra tokia pati, kaip ir 2 paveiksle pavaizduoto šablono, o skiriasi tik jų taikymo sritis, naudojami algoritmai, įėjimai ir išėjimai.

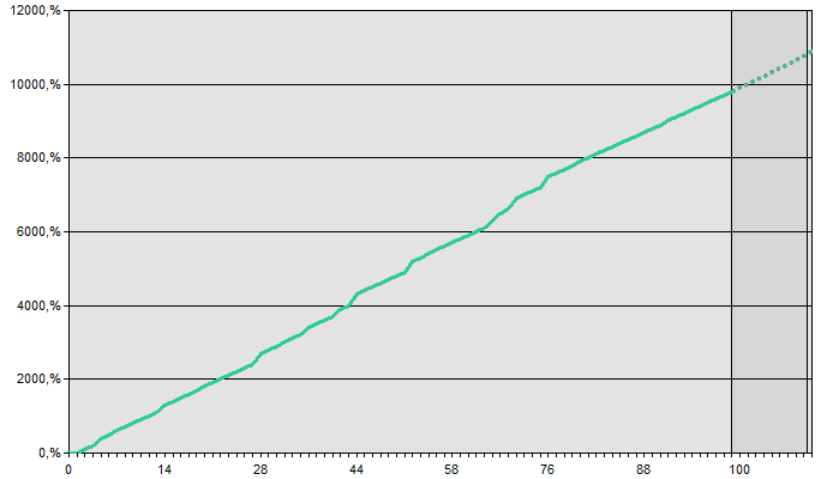
Dažniausiai, vieno tipo šabloną galima pritaikyti panašiems uždaviniam spręsti [7]. Kaip pavyzdį paimsime žmonių analizės sistemą, kurioje saugomi skirtingų pasaulio regiono žmonių fizinių parametrų duomenys (ūgis, kūno sudėjimas, amžius). Tokia sistema leistų analizuoti žmonių fizinius parametrus pagal regionus ir kaip tie parametrai keičiasi laikui bėgant. Tokiam uždaviniui pritaikius miškų augimo prognozei sudarytą šabloną, kurio schema pavaizduota 2 paveiksle, galima būtų prognozuoti žmonių fizinius parametrus skirtinguose regionuose. Šablonams, kurie naudoja kitus algoritmus, galioja ta pati taisyklė - vieną šabloną galima taikyti keliems uždaviniams. Pavyzdžiui, šabloną, naudojantį sprendimo medžių algoritmą galima pritaikyti užduočiai, kai reikia priimti sprendimą į kurį produktą geriausiai investuoti pinigus. Toks uždavinys gali pasitaikyti daugelyje gamybos sričių, o uždavinio sprendimo įėjimai ir išėjimai bus vienodi.

Prieš pradedant kurti duomenų gavybos šabloną, reikia gerai ištirti uždavinį, ir suprasti jo tikslus. Tai padės pasirinkti uždavinio sprendimui tinkamą algoritmą. Gali atsitikti taip, kad uždavinio sprendimui tinka keli algoritmai. Tada galima sukurti kelis šablonus ir šablono derinimo eigoje išsirinkti geriausius rezultatus parodantį šabloną.

Šablono derinimas atliekamas paduodant jam daugiamačių duomenų pjūvius ir nagrinėjant gautus rezultatus. Tokias operacijas reikia atlikti kelis kartus, kiekvieną katrą paduodant skirtingus duomenų pjūvius. Derinimo metu šablonas įsimena analizuojamų duomenų savybes, kurias vėliau galės

taikyti analizuojant realius duomenis. Kadangi duomenis pastoviai keičiasi, tai šablono derinimą reikia daryti reguliariai. Kartais naudinga išvalyti visas šablono atmintyje saugomas duomenų savybes ir iš naujo suderinti jau tuščią šabloną. Derinimo rezultatus kartais gali būti labai sudėtinga suprasti. Taip gali atsitikti dėl kelių priežasčių. Kartais duomenys gali turėti labai aukštą atsitiktinumo lygį ir neturėti jokių dėsningumų, nors realūs duomenis dažniausiai juos turi. Kitas atvejis gali būti toks, kad šablonui paduodami netinkamo formato duomenis. Tada tikslinga būtų peržiūrėti duomenų transformavimo ir išgryninimo procedūras ir pagerinti parametrų kokybę [3].

Aprašyto šablono testavimo rezultatai pateikiami 3 paveiksle. Diagrama parodo medžių augimą procentais, priklausomai nuo amžiaus. Dešinėje grafiko pusėje, punktyrine linija pavaizduota medžio augimo prognozė dešimčiai metų į priekį. Šablono algoritmas akivaizdžiai atrado medžių augimo koeficientus ir suprognozavo, kad medžių augimas išliks tolygus.



3 pav. Miško augimo prognozavimo testo pavyzdys

Grįžkime prie Lietuvos miškų informacinės sistemos. Faktiniai duomenys apie medžių augimą įvairiose vietovėse renkami kas penkeri metai, o kirtimai gali būti vykdomi kiekvienais metais, reiškia duomenų saugykloje atsiranda duomenų trūkumas. DMX šabloną galima integruoti į ETL procesą ir saugyklą pildyti duomenimis iš tos pačios saugyklos pritaikius duomenų prognozavimo šabloną. Tai gali išspręsti duomenų trūkumo problemą ir leis naudoti nuspėjamus duomenis medžių kirtimo veikloje. Faktinė informacija, kuri surinkinėjama kas penkeri metai sumažina bendros informacijos paklaidas ir yra pagrindinė informacija, pagal kurią galima daryti augimo prognozes.

3 Išvados

Užklausų kalba MDX skirta darbui su daugiamačiais duomenimis, todėl ji pralenkia SQL užklausų kalbą šitoje srityje. Taip pat egzistuoja DMX kalba, skirta duomenų gavybai, kuri gali būti naudojama tiek su reliaciniais, tiek su daugiamačiais duomenimis. Remiantis tokiais faktais galima teigti, jog daugiamačių duomenų analizei geriausiai naudoti MDX ir DMX kalbų derinį.

MDX ir DMX kalbų taikymas gali padidinti veiklos analizės efektyvumą. Tai rodo duomenų prognozavimui sudarytas DMX šablonas, kuris išbandytas su pavyzdiniais duomenimis miškų informacijos prognozavimui. Naudojant vien tik SQL, prognozavimui realizuoti reikėtų kurti sudėtingus algoritmus.

Sprendžiant duomenų trūkumo problema, MDX šabloną galima integruoti į ETL procesą, kas leidžia papildyti duomenų saugyklą pagal turimus duomenis prognozuojamais duomenimis.

Literatūros sąrašas

[1] **A. Silverstein, C. Gallelli, P. Bertucci, R. Rankins**. Microsoft SQL Server 2005 Unleashed. *Sams Publishing*, 2007, 1484.

[2] **E. Thomson**. OLAP Solutions. Building Multidimensional Information Systems. *John Wiley & Sons, Inc*., 2002, 107 – 134.

[3] **J. MacLennan, Z. Tang**. Data Mining With SQL Server 2005. *Wiley Publishing, Inc.,* 2005, 43, 296 - 297.

[4] **L. Langit**. Foundations of SQL Server 2005 Business Intelligence. *Apress,* 2007, 246 – 247.

[5] **M. Ross, R. Kimball**. The Data Warehouse Toolkit. The Complete Guide to dimensional Modeling. *John Wiley & Sons, Inc*., 2002, 343 – 345.

[6] **S. Harinath, S. Quinn**. Professional SQL Server Analysis Services 2005 with MDX. *Wrox Press*, 2006.

[7] **P. Smith, G. Renganathan, G. Das, K. Lin, H. Manila**. Rule Discovery From Time Series. http://citeseer.ist.psu.edu/das98rule.html, 1998, žiūrėta: 2008.03.12.

Multidimensional data analysis extension with data mining patterns

In this paper we will discuss an approach to increase the efficiency of data analysis by using multidimensional data query and analysis languages like MDX and DMX. These languages are poorly known and have great potential. For example we can enhance dada warehouse queries with Data Mining functions, which can give us very quick and accurate results. In this paper we will also discuss possibilities to use DMX patterns for data prediction.