

Table of Contents

- 1. Подготовка данных
 - 1.1. Библиотеки и настройки
 - 1.2. Создание датафрейма
 - 1.2.1. Чтение файла и запись данных в датафрейм
 - 1.2.2. Разбиение данных в датафрейме на столбцы
 - 1.2.3. Добавление дополнительных столбцов для анализа, реиндексация
- 2. Статистика сообщений
 - 2.1. Сводные показатели
 - 2.2. Группировки и распределение
 - 2.2.1. Распределение количества сообщений и участников
 - 2.2.2. Динамика активности в чате
 - 2.2.3. Активность в чате по времени суток и по дням недели
- 3. Анализ содержания сообщений
 - 3.1. Лексический анализ
 - 3.1.1. Частотные характеристики словаупотребления
 - 3.1.1.1 Распределение слов по частотным характеристикам
 - 3.1.1.2 Характеристика сообщений по средней частоте встречаемости используемых слов
 - 3.2. Анализ смысловой окрашенности сообщений
 - 3.2.1. Создание перечней позитивно и негативно окрашенных слов
 - 3.2.2. Распределение позитивных и негативных слов по частотным характеристикам
 - 3.2.3. Характеристика сообщений по степени позитивности и негативности
 - 3.2.4. Группировка участников по степени смысловой окрашенности сообщений
- 4. Анализ актуальных тем
 - 4.1. Составление словаря актуальных тем и ключевых слов.
 - 4.2. Частота упоминаний ключевых слов и актуальных тем в сообщениях
 - 4.3. Количество актуальных тем по каждому заинтересованному участнику
 - 4.4. Количество участников по каждой актуальной теме
 - 4.5. Динамика обсуждения актуальных тем
- 5. Определение доверительного интервала для заинтересованности членов СНТ в актуальных темах с уровнем доверия 0,95 (95%)
 - 5.1. Расчёт погрешности
 - 5.2. Доверительные интервалы для заинтересованности всех членов СНТ
 - 5.3. Доверительный интервал для доли активных членов СНТ (готовых принимать участие в обсуждении актуальных тем)
 - 5.4. Доверительные интервалы для заинтересованности активных членов СНТ
- 6. ВЫВОДЫ

1. Подготовка данных

1.1. Библиотеки и настройки

```
In [1]: import numpy as np
import pandas as pd
import scipy.stats as st
import math

import matplotlib as mpl
import matplotlib.pyplot as plt
import seaborn as sns
import seaborn.objects as so

import re
import pymorphy2
from bs4 import BeautifulSoup
import requests

import os
import joblib
```

```
In [2]: # configure multiple output
from IPython.core.interactiveshell import InteractiveShell
InteractiveShell.ast_node_interactivity = "all"

# set DF columns and rows maxima
pd.set_option('display.max_rows', 400)
pd.set_option('display.max_columns', 50)

# reset sns defaults
dummy = sns.reset_defaults
sns.set_theme(style="darkgrid")
```

1.2. Создание датафрейма

1.2.1. Чтение файла и запись данных в датафрейм

The structure of WhatsApp txt file contains not a csv. It may also contain multiple lines in messages. Let's read the file one line at a time and combine the dataframe date column and message column using regular expressions and python. We shall perform further dataframe purifying using pandas methods.

```
In [3]: # Needed regular expressions

regex_date = re.compile(r'\d\d\.\d\d\.\d\d\d\d')
regex_time = re.compile(r'\d\d:\d\d')
regex_datetime = re.compile(r'\d{2}\.\d{2}\.\d{4}, \d{2}:\d{2}')
regex_user1 = re.compile(r'((?<-- ).+?(?=:))')
regex_user2 = re.compile(r'((?<-- ).+(?=\\s\\b\\S+\\())')
```

```
In [4]: # Chat dataframe
df_chat = pd.DataFrame(columns=['raw_text'])
```

```
In [5]: # read the file line-by-line, combine multiple lines into one, append raw text to a single column
msg_text = ''
splitter = None
with open("data/2023-04-14_chat.txt", mode='r', encoding='UTF-8') as file_in:
    for msg_line in file_in: # reading file line at a time

        splitter = re.search(regex_datetime, msg_line) # Looking for date and time
        if splitter != None: # date and time found: end the previous message and start a new one
            if msg_text != '': # if there is a previous message, write it to dataframe
                ser_new_msg = pd.Series({'raw_text': msg_text}) # create a series
                df_chat = pd.concat([df_chat, ser_new_msg.to_frame().T], ignore_index=True)
                msg_text = '' # delete the previous message
            msg_text = msg_line.strip() # start creating a new message, deleting spaces
        else:
            msg_text = f"{msg_text} {msg_line.strip()}" # add the new line to the previous message
df_chat = pd.concat([df_chat, ser_new_msg.to_frame().T], ignore_index=True)
```

```
In [6]: df_chat.head()
```

```
Out[6]: raw_text
0 10.09.2019, 10:53 - Сообщения и звонки защищен...
1 10.09.2019, 10:53 - +7 905 500-38-87 создал(-...
2 10.09.2019, 10:53 - +7 905 500-38-87 добавил(....
3 10.09.2019, 11:50 - +7 905 500-38-87 добавил(....
4 10.09.2019, 11:50 - +7 905 500-38-87: Здравств...
```

1.2.2. Разбиение данных в датафрейме на столбцы

Выделим дату и время

```
In [7]: # combine f-string of date and time in raw text and parse it as pd.datetime
df_chat = df_chat.assign(
    date_time=df_chat.apply(
        lambda x: pd.to_datetime(f'{x.raw_text[:10]} {x.raw_text[12:18]}',
                                dayfirst=True),
        axis=1
    )
```

Выделим имя участника и сообщения или действия:

1. Участник: текст сообщения
2. Участник сделал(-а) действие
3. Системные сообщения от имени WhatsApp

```
In [8]: # NORMAL MESSAGE TEXT
# Extract user name from normal message, separated by ':'
df_chat = (
    df_chat
    .assign(user=df_chat.raw_text.str.extract(regex_user1))
)

# Extract normal message text (after ':')
df_chat = (
    df_chat
```

```

        .assign(message=df_chat[~pd.isna(df_chat.user)].apply(
            lambda x: x.loc["raw_text"][20+len(x.loc["user"])+1:].strip(),
            axis=1
        )
    )
df_chat.head(12)

```

Out[8]:

	raw_text	date_time	user	message
0	10.09.2019, 10:53 - Сообщения и звонки защищены...	2019-09-10 10:53:00	NaN	NaN
1	10.09.2019, 10:53 - +7 905 500-38-87 создал(...	2019-09-10 10:53:00	NaN	NaN
2	10.09.2019, 10:53 - +7 905 500-38-87 добавил(...	2019-09-10 10:53:00	NaN	NaN
3	10.09.2019, 11:50 - +7 905 500-38-87 добавил(...	2019-09-10 11:50:00	NaN	NaN
4	10.09.2019, 11:50 - +7 905 500-38-87: Здравств...	2019-09-10 11:50:00	+7 905 500-38-87	Здравствуйте! Данная группа для общения жите...
5	10.09.2019, 11:57 - +7 905 500-38-87 добавил(...	2019-09-10 11:57:00	NaN	NaN
6	10.09.2019, 12:13 - +7 977 937-32-70: Хорошая ...	2019-09-10 12:13:00	+7 977 937-32-70	Хорошая идея!
7	10.09.2019, 12:14 - +7 916 300-00-18:	2019-09-10 12:14:00	+7 916 300-00-18	
8	10.09.2019, 12:14 - +7 926 248-32-75: Да, согл...	2019-09-10 12:14:00	+7 926 248-32-75	Да, согласна.
9	10.09.2019, 12:14 - +7 985 956-97-17: Поддерживаю	2019-09-10 12:14:00	+7 985 956-97-17	Поддерживаю
10	10.09.2019, 12:15 - +7 926 248-32-75: У нас во...	2019-09-10 12:15:00	+7 926 248-32-75	У нас вопрос по поводу асфальта, председатель ...
11	10.09.2019, 12:16 - +7 926 248-32-75: Наверху ...	2019-09-10 12:16:00	+7 926 248-32-75	Наверху сделали и прямо красота.

In [9]:

```

# USER ACTIONS
# Extract user name from user actions - no separation by containing '(-a)'
df_chat[pd.isna(df_chat.user)] = (
    df_chat[pd.isna(df_chat.user)]
    .assign(user=df_chat.raw_text.str.extract(regex_user2))
)

# Mask for selecting rows with NaN in message and not NaN in user
mask = pd.isna(df_chat.message) & ~pd.isna(df_chat.user)

# Extract action (no ":" separator)
df_chat =(
    df_chat
    .assign(action=df_chat[mask].apply(
        lambda x: x.loc["raw_text"][20+len(x.loc["user"]):].strip(),
        axis=1
    )
)
)

```

In [10]: df_chat.head(12)

	raw_text	date_time	user	message	action
0	10.09.2019, 10:53 - Сообщения и звонки защищены...	2019-09-10 10:53:00	NaN	NaN	NaN
1	10.09.2019, 10:53 - +7 905 500-38-87 создал(-...)	2019-09-10 10:53:00	+7 905 500-38-87	NaN	создал(-а) группу "СПК "Ветераны Победы""
2	10.09.2019, 10:53 - +7 905 500-38-87 добавил(...)	2019-09-10 10:53:00	+7 905 500-38-87	NaN	добавил(-а) вас
3	10.09.2019, 11:50 - +7 905 500-38-87 добавил(...)	2019-09-10 11:50:00	+7 905 500-38-87	NaN	добавил(-а) контакт +7 916 600-01-65, +7 985 4...
4	10.09.2019, 11:50 - +7 905 500-38-87: Здравств...	2019-09-10 11:50:00	+7 905 500-38-87	Здравствуйте! Данная группа для общения житеle...	NaN
5	10.09.2019, 11:57 - +7 905 500-38-87 добавил(...)	2019-09-10 11:57:00	+7 905 500-38-87	NaN	добавил(-а) контакт +7 926 565-55-49
6	10.09.2019, 12:13 - +7 977 937-32-70: Хорошая ...	2019-09-10 12:13:00	+7 977 937-32-70	Хорошая идея!	NaN
7	10.09.2019, 12:14 - +7 916 300-00-18: 👍	2019-09-10 12:14:00	+7 916 300-00-18	👍	NaN
8	10.09.2019, 12:14 - +7 926 248-32-75: Да, согл...	2019-09-10 12:14:00	+7 926 248-32-75	Да, согласна.	NaN
9	10.09.2019, 12:14 - +7 985 956-97-17: Поддерживаю	2019-09-10 12:14:00	+7 985 956-97-17	Поддерживаю	NaN
10	10.09.2019, 12:15 - +7 926 248-32-75: У нас во...	2019-09-10 12:15:00	+7 926 248-32-75	У нас вопрос по поводу асфальта, председатель ...	NaN
11	10.09.2019, 12:16 - +7 926 248-32-75: Наверху ...	2019-09-10 12:16:00	+7 926 248-32-75	Наверху сделали и прямо красота.	NaN

In [11]: # SYSTEM MESSAGES

mask for NaN in user and NaN in message

mask = pd.isna(df_chat.user) & pd.isna(df_chat.message)

Extract system messages

```
df_chat[mask] = (
    df_chat[mask]
    .assign(action=df_chat[mask].apply(
        lambda x: x.loc["raw_text"][:20].strip(),
        axis=1
    )
)
```

```
# Replace remaining NaN in user with "WhatsApp" - system information
df_chat.user.fillna("WhatsApp", inplace=True)
```

In [12]: df_chat.head(12)

	raw_text	date_time	user	message	action
0	10.09.2019, 10:53 - Сообщения и звонки защищены сквозным шифрованием...	2019-09-10 10:53:00	WhatsApp	NaN	Сообщения и звонки защищены сквозным шифрованием...
1	10.09.2019, 10:53 - +7 905 500-38-87 создал(-а) группу "СПК "Ветераны Победы""	2019-09-10 10:53:00	+7 905 500-38-87	NaN	создал(-а) группу "СПК "Ветераны Победы""
2	10.09.2019, 10:53 - +7 905 500-38-87 добавил(-а) вас	2019-09-10 10:53:00	+7 905 500-38-87	NaN	добавил(-а) вас
3	10.09.2019, 11:50 - +7 905 500-38-87 добавил(-а) контакт +7 916 600-01-65, +7 985 4...	2019-09-10 11:50:00	+7 905 500-38-87	NaN	добавил(-а) контакт +7 916 600-01-65, +7 985 4...
4	10.09.2019, 11:50 - +7 905 500-38-87: Здравствуйте! Данная группа для общения жите...	2019-09-10 11:50:00	+7 905 500-38-87	Здравствуйте! Данная группа для общения жите...	NaN
5	10.09.2019, 11:57 - +7 905 500-38-87 добавил(-а) контакт +7 926 565-55-49	2019-09-10 11:57:00	+7 905 500-38-87	NaN	добавил(-а) контакт +7 926 565-55-49
6	10.09.2019, 12:13 - +7 977 937-32-70: Хорошая идея!	2019-09-10 12:13:00	+7 977 937-32-70	Хорошая идея!	NaN
7	10.09.2019, 12:14 - +7 916 300-00-18: 👍	2019-09-10 12:14:00	+7 916 300-00-18	👍	NaN
8	10.09.2019, 12:14 - +7 926 248-32-75: Да, согла...	2019-09-10 12:14:00	+7 926 248-32-75	Да, согласна.	NaN
9	10.09.2019, 12:14 - +7 985 956-97-17: Поддерживаю	2019-09-10 12:14:00	+7 985 956-97-17	Поддерживаю	NaN
10	10.09.2019, 12:15 - +7 926 248-32-75: У нас вопрос по поводу асфальта, председатель ...	2019-09-10 12:15:00	+7 926 248-32-75	У нас вопрос по поводу асфальта, председатель ...	NaN
11	10.09.2019, 12:16 - +7 926 248-32-75: Наверху ...	2019-09-10 12:16:00	+7 926 248-32-75	Наверху сделали и прямо красота.	NaN

Удалим столбец raw_text - все его данные распределены

In [13]: df_chat.drop(columns=['raw_text'], inplace=True)

In [14]: df_chat.sample(12)

Out[14]:

	date_time	user	message	action
2968	2022-03-01 16:53:00	+7 915 002-59-99	Можно конечно. Кто попросит?	NaN
2165	2021-09-19 12:10:00	+7 905 500-38-87	Та не надо мне делать нерви, их есть кому порт...	NaN
3404	2022-07-05 09:09:00	+7 915 002-53-33	<Без медиафайлов>	NaN
2010	2021-06-07 20:59:00	+7 926 582-05-88	Всеволод 😊	NaN
3592	2022-08-08 11:22:00	+7 920 213-01-91	Да им по ходу всё до звезды	NaN
3935	2022-09-22 11:10:00	+7 985 273-58-42	<Без медиафайлов>	NaN
2996	2022-03-10 09:17:00	+7 926 088-22-70	9 мая нет	NaN
3727	2022-08-25 10:09:00	+7 985 273-58-42	Доброе утро! Повестка дня обозначена в объявл...	NaN
2988	2022-03-09 20:20:00	+7 926 565-55-49	Соседи! У кого-нибудь есть опрессовщик сегодня...	NaN
4269	2022-10-31 11:36:00	+7 903 763-79-96	Саморазбор и самовынос 😊	NaN
6037	2023-04-04 10:29:00	+7 926 667-77-23	В пульте мы все белоручки.	NaN
1873	2021-03-18 06:14:00	+7 905 500-38-87	Скорую видел не раз у нас здесь. Набери 112	NaN

1.2.3. Добавление дополнительных столбцов для анализа, реиндексация

In [15]:

```
# Day of week

# Day of week to verbose
dict_days_of_week = {
    '0': 'понедельник',
    '1': 'вторник',
    '2': 'среда',
    '3': 'четверг',
    '4': 'пятница',
    '5': 'суббота',
    '6': 'воскресенье'
}
df_chat = (
    df_chat
    .assign(
        day_of_week=df_chat.date_time
        .apply(lambda x: x.dayofweek)
    )
)
df_chat = (
    df_chat
    .assign(
        DOW_name=df_chat.date_time
        .apply(lambda x: dict_days_of_week[str(x.dayofweek)]))
)
```

```
)
```

```
In [16]: # Hour
df_chat = (
    df_chat
    .assign(
        hour=df_chat.date_time
        .apply(lambda x: x.hour)
    )
)
```

```
In [17]: # 'date_time' as index
df_chat = df_chat.set_index('date_time')
df_chat.sample(12)
```

Out[17]:

	user	message	action	day_of_week	DOW_name	hour
date_time						
2020-12-18 14:49:00	+7 985 444-72-00	Что обсуждать. Надо идти и смотреть. Сейчас та...	NaN	4	пятница	14
2020-05-11 11:14:00	+7 985 222-38-78	А сколько у нас домовладений в спк?	NaN	0	понедельник	11
2023-01-07 15:21:00	+7 965 280-67-40	<Без медиафайлов>	NaN	5	суббота	15
2021-01-12 13:45:00	+7 977 839-40-24	Заработал	NaN	1	вторник	13
2022-12-01 14:16:00	+7 926 226-65-66	<Без медиафайлов>	NaN	3	чеверг	14
2022-12-30 17:09:00	+7 926 180-22-18	Оно	NaN	4	пятница	17
2022-02-25 13:19:00	+7 920 213-01-91	Данное сообщение удалено	NaN	4	пятница	13
2020-12-11 18:41:00	+7 903 222-18-60	Надо будет нам переделать...	NaN	4	пятница	18
2022-08-17 08:43:00	+7 985 273-58-42	<Без медиафайлов>	NaN	2	среда	8
2020-08-04 20:50:00	+7 985 710-78-10	Я на Кубани буду примерно к выходным, загрузил...	NaN	1	вторник	20
2023-02-25 19:46:00	+7 926 180-22-18	Будьте внимательны	NaN	5	суббота	19
2022-07-11 00:17:00	+7 926 226-65-66	Снова	NaN	0	понедельник	0

2. Статистика сообщений

2.1. Сводные показатели

Общая информация о DF

In [18]:

```
df_chat.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 6301 entries, 2019-09-10 10:53:00 to 2023-04-14 18:48:00
Data columns (total 6 columns):
 #   Column      Non-Null Count  Dtype  
--- 
 0   user        6301 non-null   object  
 1   message     6131 non-null   object  
 2   action       170 non-null   object  
 3   day_of_week 6301 non-null   int64  
 4   DOW_name    6301 non-null   object  
 5   hour        6301 non-null   int64  
dtypes: int64(2), object(4)
memory usage: 344.6+ KB
```

```
In [19]: n_messages = df_chat.message.count()
n_users = df_chat.user.nunique()
df_chat_tmp = (
    df_chat[['user', 'message']]
    .groupby('user', as_index=False)
    .agg('count')
)
n_active_users = df_chat_tmp[df_chat_tmp.message > 2].user.count()

print(f"Общее количество сообщений: {n_messages}")
print(f"Число участников: {n_users}")
print(f"Число активных участников (более 2 сообщений): {n_active_users}")
```

```
Общее количество сообщений: 6131
Число участников: 227
Число активных участников (более 2 сообщений): 97
```

2.2. Группировки и распределение

2.2.1. Распределение количества сообщений и участников

```
In [20]: # calculating 0.9 quantile
q09 = df_chat[['user', 'message']].groupby('user').agg('count').quantile(q=0.9)[0]

print(f"Наиболее активные участники чата (топ 10%): количество сообщений от одного")

df_chat_tmp = (
    df_chat[['user', 'message']]
    .groupby('user')
    .agg('count')
    .sort_values(by='message', ascending=False)
)
df_chat_tmp[df_chat_tmp.message > q09]
print(f"Количество наиболее активных участников (свыше 90-го процентиля): "
      f"{len(df_chat_tmp[df_chat_tmp.message > q09])}\n"
      f"Что составляет {len(df_chat_tmp[df_chat_tmp.message > q09])/len(df_chat_tm"
      f" всех участников")}
```

```
Наиболее активные участники чата (топ 10%): количество сообщений от одного участни
ка более 86
```

Out[20]:

user	message
+7 985 769-03-22	599
+7 985 273-58-42	410
+7 985 444-72-00	340
+7 905 500-38-87	297
+7 925 066-47-95	290
+7 915 002-53-33	179
+7 903 222-18-60	153
+7 920 213-01-91	151
+7 916 149-49-26	137
+7 929 552-10-20	132
+7 926 180-22-18	128
+7 926 088-22-70	125
+7 926 112-00-22	123
+7 926 278-42-47	121
+7 915 134-67-88	118
+7 985 222-38-78	114
+7 903 578-91-95	111
+7 915 449-73-74	111
+7 926 226-65-66	109
+7 977 322-86-98	108
+7 915 002-59-99	97
+7 926 248-32-75	96
+7 926 667-77-23	88

Количество наиболее активных участников (свыше 90-го процентиля): 23
Что составляет 10% всех участников

In [21]:

```
data2211=df_chat[['user', 'message']].groupby('user').agg('count')

fig, ax = plt.subplots(figsize=(14, 7))

g = sns.histplot(
    data=data2211,
    x='message',
    ax=ax
)
plt.yscale('log')

# Mean Line
mean_value = data2211.message.mean()
dummy = ax.axvline(x=mean_value, color="red")
# mean line annotation
dummy = ax.annotate(
    text=f"Средняя {mean_value:.0f}",
```

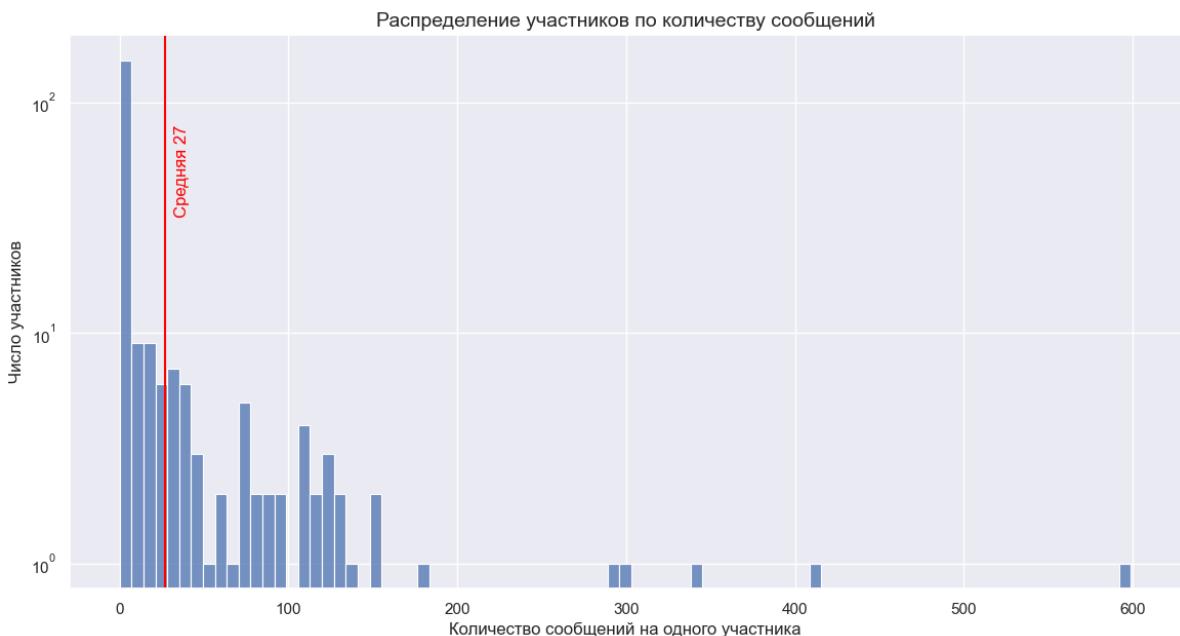
```

        xy=(mean_value+10, 50), # text position
        xycoords='data',
        horizontalalignment='center',
        verticalalignment='center',
        rotation=90, # Rotation
        color='red', # Color
        alpha=1,
        fontsize=12,
        fontweight="normal")

dummy = plt.title('Распределение участников по количеству сообщений', fontsize=14)
dummy = plt.xlabel('Количество сообщений на одного участника')
dummy = plt.ylabel('Число участников')

plt.show()

```



Описание этого распределения по процентилям

```
In [22]: #Quantiles
print("КВАНТИЛИ (ПРОЦЕНТИЛИ)")
arr_q_tiles = np.percentile(data2211, np.arange(10, 101, 10))
for i, n_msgs in enumerate(arr_q_tiles):
    print(f"- {(i+1)/10:.0%} участников: не более {n_msgs:.0f} сообщений от одного человека")
print(f"Медиана (50-й процентиль): 50% участников написале не более {data2211.message.nunique()}")
f"сообщений от одного человека")
```

КВАНТИЛИ (ПРОЦЕНТИЛИ)

- 10% участников: не более 0 сообщений от одного человека
 - 20% участников: не более 0 сообщений от одного человека
 - 30% участников: не более 0 сообщений от одного человека
 - 40% участников: не более 0 сообщений от одного человека
 - 50% участников: не более 0 сообщений от одного человека
 - 60% участников: не более 4 сообщений от одного человека
 - 70% участников: не более 12 сообщений от одного человека
 - 80% участников: не более 34 сообщений от одного человека
 - 90% участников: не более 86 сообщений от одного человека
 - 100% участников: не более 599 сообщений от одного человека
- Медиана (50-й процентиль): 50% участников написале не более 0 сообщений от одного человека

2.2.2. Динамика активности в чате

У нас нет данных о времени прочтения участниками сообщений в чате. Поэтому, единственным имеющимся способом измерить активность чата может быть количество написанных сообщений на момент времени.

```
In [23]: # timespan
n_days = int((df_chat.index.max() - df_chat.index.min()) / np.timedelta64(1, 'D'))
n_weeks = n_days / 7
n_years = n_days / 365
n_hours = n_days * 24
print(f"В чате записаны сообщения за {n_days} дней, что составляет:\n"
      f"{n_weeks:.1f} недель\n"
      f"{n_years:.1f} лет\n"
      f"{n_hours} часов")
```

В чате записаны сообщения за 1312 дней, что составляет:
187.4 недель
3.6 лет
31488 часов

```
In [24]: data2221 = df_chat.message.resample('M').count()

g = sns.relplot(data=data2221,
#                  x=data2221.index.date,
#                  y='message',
#                  height=4,
#                  aspect=3.2,
#                  marker='o',
#                  color='darkgreen',
#                  kind='line',
#                  )
# setting x scale
scale_span_x = pd.date_range(start=data2221.index.min(),
                             end=data2221.index.max(),
                             freq='M').tolist()
dummy = g.set(xticks=scale_span_x)
dummy = g.set_xticklabels(rotation=90)
dummy = g.set_xlabels('Итоговый день месяца', fontsize=12)
dummy = g.set_ylabels('Количество сообщений', fontsize=12)

# mean value line
mean_value = data2221.mean()
# drawing h line
dummy = g.ax.axhline(mean_value, color='black', linewidth=2, linestyle=':')
dummy = g.ax.annotate(
    text=f"Среднемесячное количество сообщений: {(mean_value):.0f}",
    xy=(0.17, 0.25), # text position
    xycoords="axes fraction",
    horizontalalignment='center',
    verticalalignment='center',
    rotation=0, # Rotation
    color='black', # Color
    alpha=1,
    fontsize=12,
    fontweight="normal")

dummy = plt.suptitle("Динамика месячного количества сообщений в чате", y=1.025)
plt.show()
```



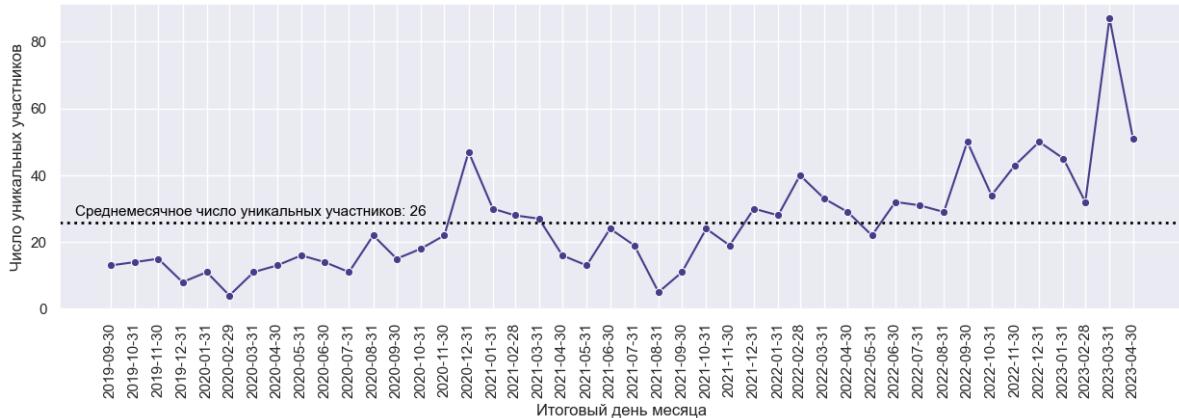
```
In [25]: data2222 = df_chat.user.resample('M').nunique()

g = sns.relplot(data=data2222,
                 height=4,
                 aspect=3.2,
                 marker='o',
                 color='darkslateblue',
                 kind='line',
                 )
# setting x scale
scale_span_x = pd.date_range(start=data2222.index.min(),
                               end=data2222.index.max(),
                               freq='M').tolist()
dummy = g.set(xticks=scale_span_x)
dummy = g.set_xticklabels(rotation=90)
dummy = g.set_xlabels('Итоговый день месяца', fontsize=12)
dummy = g.set_ylabels('Число уникальных участников', fontsize=12)

# mean value line
mean_value=data2222.mean()
# drawing h line
dummy = g.ax.axhline(mean_value, color='black', linewidth=2, linestyle=':')
dummy = g.ax.annotate(
    text=f"Среднемесячное число уникальных участников: {(mean_value):.0f}",
    xy=(0.17, 0.32), # text position
    xycoords="axes fraction",
    horizontalalignment='center',
    verticalalignment='center',
    rotation=0, # Rotation
    color='black', # Color
    alpha=1,
    fontsize=11,
    fontweight="normal")

dummy = plt.suptitle("Динамика ежемесячного числа уникальных участников чата", y=1
plt.show()
```

Динамика ежемесячного числа уникальных участников чата



```
In [26]: #data2223 = df_chat.user.resample('M').nunique()
data2223 = (
    df_chat[['user', 'message']]
    .resample('M')
    .nunique()
)
data2223 = (
    data2223
    .assign(msg_per_user=data2223.message / data2223.user)
)

g = sns.relplot(data=data2223.msg_per_user,
                 height=4,
                 aspect=3.2,
                 marker='o',
                 color='darkorange',
                 kind='line',
                 )
# setting x scale
scale_span_x = pd.date_range(start=data2223.index.min(),
                               end=data2223.index.max(),
                               freq='M').tolist()
dummy = g.set(xticks=scale_span_x)
dummy = g.set_xticklabels(rotation=90)
dummy = g.set_xlabels('Итоговый день месяца', fontsize=12)
dummy = g.set_ylabels(f"Среднее количество сообщений \nна одного уникального участника", fontsize=12)

# mean value line
mean_value=data2223.msg_per_user.mean()
# drawing h line
dummy = g.ax.axhline(mean_value, color='black', linewidth=2, linestyle=':')
dummy = g.ax.annotate(
    text=f"Средняя: {mean_value:.0f}",
    xy=(0.11, 0.3), # text position
    xycoords="axes fraction",
    horizontalalignment='center',
    verticalalignment='center',
    rotation=0, # Rotation
    color='black', # Color
    alpha=1,
    fontsize=12,
    fontweight="normal")

dummy = plt.suptitle("Динамика среднемесячного количества сообщений на одного уник")
plt.show()
```



2.2.3. Активность в чате по времени суток и по дням недели

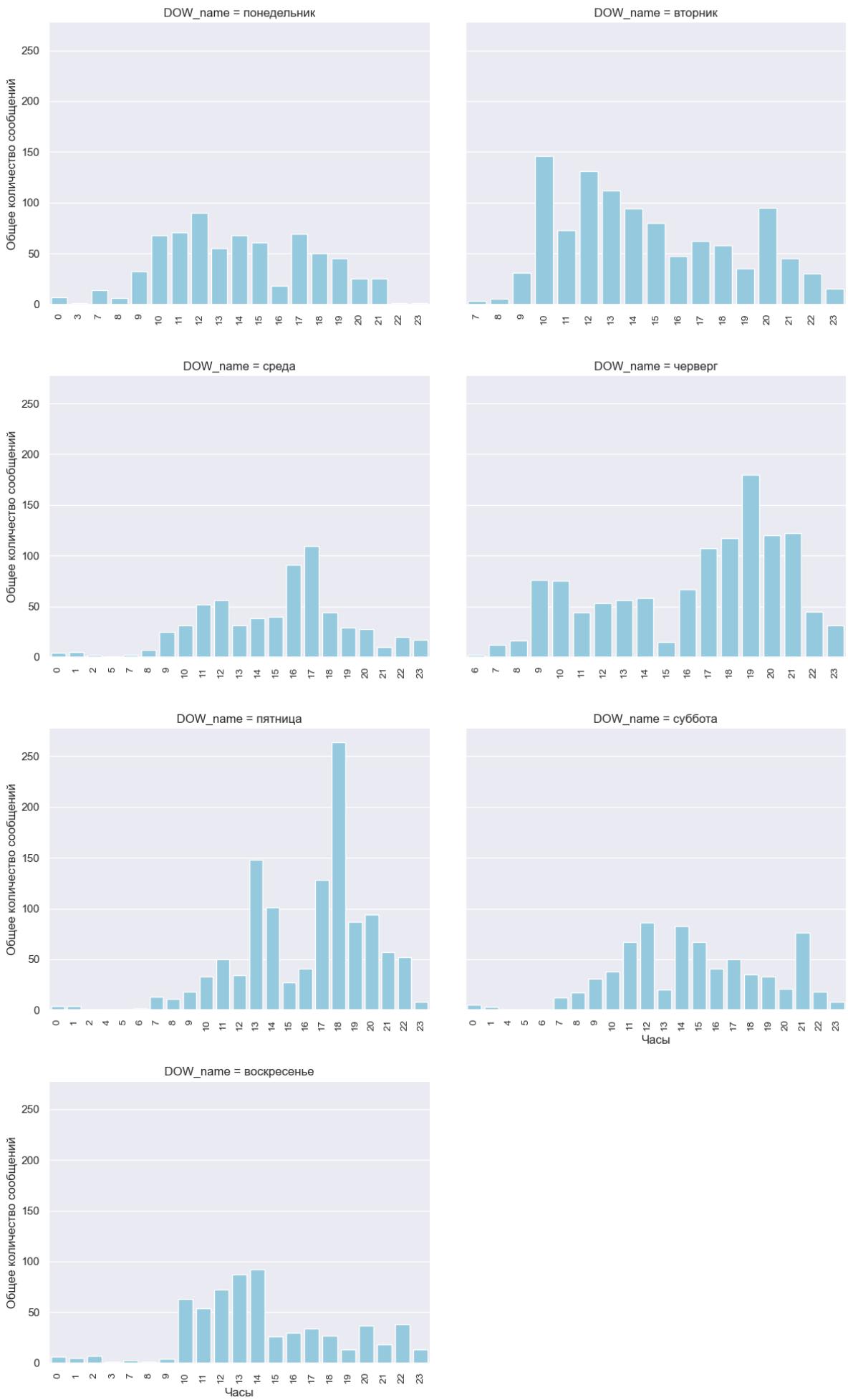
```
In [27]: data2231 = (
    df_chat[['message', 'day_of_week', 'DOW_name', 'hour']]
    .groupby(['day_of_week', 'DOW_name', 'hour'], as_index=False)
    .agg('count')
)

g=sns.catplot(data=data2231,
               x='hour',
               y= 'message',
               col='DOW_name',
               col_wrap=2,
               kind='bar',
               errorbar=('ci', False),
               dodge=True,
               sharex=False,
               sharey=True,
               color='skyblue',
               height=5,
               aspect=1.2,
               )

dummy = plt.suptitle("Распределение общего количества сообщений в чате по часам и дням недели", fontsize=14, x=0.5, y=1)
# Prarmeters:
dummy = g.set_xticklabels(rotation=90, fontsize=10)
dummy = g.set_ylabels('Общее количество сообщений', fontsize=12)
dummy = g.set_xticklabels(size=10)
dummy = g.set_xlabels('Часы', size=12)
dummy = g.set_titles(size=12)

dummy = g.tight_layout(h_pad=3,w_pad=3)
plt.show()
```

Распределение общего количества сообщений в чате по часам и по дням недели



```
In [28]: data2232 = (
    df_chat[['user', 'day_of_week', 'DOW_name', 'hour']]
    .groupby(['day_of_week', 'DOW_name', 'hour'], as_index=False)
    .agg('nunique')
)

g=sns.catplot(data=data2232,
                x='hour',
                y= 'user',
                col='DOW_name',
                col_wrap=2,
                kind='bar',
                errorbar=('ci', False),
                dodge=True,
                sharex=False,
                sharey=True,
                color='lightgreen',
                height=5,
                aspect=1.2,
                )

dummy = plt.suptitle(f"Распределение общего количества уникальных участников, написавших по часам и по дням недели",
                      fontsize=14, x=0.5, y=1.0)
# Parameters:
dummy = g.set_xticklabels(rotation=90, fontsize=10)
dummy = g.set_ylabels('Общее количество уникальных участников', fontsize=12)
dummy = g.set_xticklabels(size=10)
dummy = g.set_xlabels('Часы', size=12)
dummy = g.set_titles(size=12)

dummy = g.tight_layout(h_pad=3,w_pad=3)
plt.show()
```

Распределение общего количества уникальных участников, написавших сообщения в чате,
по часам и по дням недели



```
In [29]: data2233 = (
    df_chat[['user', 'message', 'day_of_week', 'DOW_name', 'hour']]
    .groupby(['day_of_week', 'DOW_name', 'hour'], as_index=False)
    .agg('nunique')
)
data2233 = (
    data2233
    .assign(msg_per_user = data2233.message / data2233.user)
)

g=sns.catplot(data=data2233,
               x='hour',
               y= 'msg_per_user',
               col='DOW_name',
               col_wrap=2,
               kind='bar',
               errorbar=('ci', False),
               dodge=True,
               sharex=False,
               sharey=True,
               color='orange',
               height=5,
               aspect=1.2,
               )

dummy = plt.suptitle(f"Распределение среднего количества сообщений на одного уникального участника\nпо часам и по дням недели",
                     fontsize=14, x=0.5, y=1.0)
# Parameters:
dummy = g.set_xticklabels(rotation=90, fontsize=10)
dummy = g.set_ylabels(f'Среднее количество сообщений на \nодного уникального участника', fontsize=12)
dummy = g.set_xticklabels(size=10)
dummy = g.set_xlabels('Часы', size=12)
dummy = g.set_titles(size=12)

dummy = g.tight_layout(h_pad=3,w_pad=3)
plt.show()
```

Распределение среднего количества сообщений на одного уникального участника,
по часам и по дням недели



3. Анализ содержания сообщений

В данном разделе комментарии к коду будут, в основном, на русском языке

3.1. Лексический анализ

3.1.1. Частотные характеристики словоупотребления

```
In [30]: # ОБРАБОТКА ТЕКСТОВ ИЗ DF ДЛЯ СОЗДАНИЯ СПИСКА НОРМАЛИЗОВАННЫХ ЗНАЧИМЫХ СЛОВ

# Глобальные определения:
# Вызов метода .MorphAnalyzer() библиотеки pymorphy2:
morph = pymorphy2.MorphAnalyzer()
# Шаблон регулярного выражения для вычленения отдельных слов (любой из перечисленных)
# regex_word = re.compile(r'[a-zA-ZА-ЯА-ЯЁ]+\\b')
regex_word = re.compile(r'^\\d\\w+\\b')
# Шаблон регулярного выражения для удаления гиперссылок:
regex_no_http = re.compile(r'\\bhttp[s]?://\\S+', flags=re.MULTILINE)
# Шаблон регулярного выражения для удаления служебных маркеров WhatsApp:
regex_no_wp_arrow = re.compile(r'\\u200e')

# Список исходных текстов из DF
list_init_texts = []
# Список текстов из приведенных к исходной форме слов,
# кроме служебных частей речи и именных местоимений (список нормализованных значимых)
list_norm_clean_texts = []

# Функция для определения частей речи
def pos(word):
    """ возвращает вероятную часть речи для аргумента word """
    return morph.parse(word)[0].tag.POS

# Функция разбиения текста в список слов
def norm_clean_low_words_maker(text=""):
    """ Функция разбиения текста в список слов, приведённых к нижнему регистру,
    за исключением служебных частей речи (междометия, частицы, союзы, предлоги), а
    # Приводим исходный текст к нижнему регистру
    try: # (обработка ошибок, если в тексте попадётся NaN)
        lowered_text = text.lower()
    except:
        print('error is here!', text)

    # Удаляем из текста все гиперссылки и служебные стрелки WhatsApp:
    no_junk_text = re.sub(regex_no_http, '', lowered_text)
    # Удаляем из текста все служебные стрелки WhatsApp:
    no_junk_text = re.sub(regex_no_wp_arrow, '', no_junk_text)

    # Разбиваем текст на слова:
    # применяем метод.findall для поиска всех совпадений с шаблоном регулярного выражения
    list_words = regex_word.findall(no_junk_text)

    # Исключаем из текста междометия, частицы, союзы, предлоги, именные местоимения
    # граммы русского языка: для исключения
    # междометия, частицы, союзы, предлоги, личные местоимения
    functors_pos = {'INTJ', 'PRCL', 'CONJ', 'PREP', 'NPRO'}

    # проверяем и нормализуем слова
    list_norm_clean_low_words = [] # список нормализованных слов в нижнем регистре к
```

```

for word in list_words:
    # если результат вызова функции не является служебной частью речи или именем
    if pos(word) not in functors_pos:
        # приводим слово к исходной форме (normalized)
        parsed_word = morph.parse(word) # результат вызова morph - список из разборов
        normalized_word = parsed_word[0].normal_form # выбираем исходную форму
        # добавляем исходную форму проверенного слова в список исходных форм
        list_norm_clean_low_words.append(normalized_word) # добавляем проверенное слово
return list_norm_clean_low_words

# формируем исходный список текстов из df_chat, заменив NaN на ''
list_init_texts = df_chat.message.fillna('').tolist()

# Получим список текстов из нормализованных значимых частей речи за счёт
# применения функции norm_clean_low_words_maker к каждому элементу (тексту) в начальном списке
list_norm_clean_texts = list(map(norm_clean_low_words_maker, list_init_texts))

# результат
# print(list_norm_clean_texts)

```

In [31]:

```

# Сгенерируем для df_chat уникальные индексы сообщений msg_id
df_chat = (
    df_chat
    .assign(
        msg_id = np.arange(0, len(df_chat), 1)
    )
)
df_chat.sample(7)

```

Out[31]:

	user	message	action	day_of_week	DOW_name	hour	msg_id
date_time							
2023-02-20 10:00:00	+7 915 002- 53-33	А на какой линии?	NaN	0	понедельник	10	5041
2022-06-28 12:24:00	+7 985 444- 72-00	Вот квест у СПК сегодня))) теперь деревья разг...	NaN	1	вторник	12	3339
2023-03-28 13:38:00	+7 985 769- 03-22	Добавьте в опрос опцию " не могу ответить, нед...	NaN	1	вторник	13	5313
2022-07-10 11:43:00	+7 915 002- 53-33	Блин, это размер!!!	NaN	6	воскресенье	11	3440
2020-12-31 10:26:00	+7 985 769- 03-22	Ваши, может быть, и не будут. Но многие, к сож...	NaN	3	чертврт	10	1442
2022-09-27 13:40:00	+7 915 002- 59-99	http://knetwork.ru/	NaN	1	вторник	13	4031
2022-09-25 13:05:00	+7 929 643- 30-90	<Без медиафайлов>	NaN	6	воскресенье	13	3990

In [32]:

```
# Создадим DF с ID сообщений и соответствующими им нормализованными словами
# Создадим список ID сообщений df_chat (соответствующий индексам list_init_texts и
list_msg_idx = np.arange(0, len(df_chat), 1)

# Для создания DF из списка кортежей, определим такой список кортежей
list_of_tuples = []
for i, text in enumerate(list_norm_clean_texts): # для каждого i и текста в списке
    for word in text: # для каждого слова в таком тексте
        # соединим в кортеже ID сообщения и слово, добавим кортеж в список
        list_of_tuples.append(tuple([list_msg_idx[i], word]))

# Создадим DF на основе списка кортежей, обозначим названия столбцов
df_norm_words = pd.DataFrame(list_of_tuples, columns=['msg_id', 'words'])

df_norm_words.shape
df_norm_words.sample(7)
```

Out[32]:

(42923, 2)

Out[32]:

	msg_id	words
34127	5360	товарищество
29676	4651	хороший
9457	1401	некоторый
36344	5629	особенно
14519	2264	разводить
34094	5351	инф
14948	2314	деревня

In [33]:

```
# Создадим DF с подсчётом встречаемости нормализованных слов (по убыванию) среди всех сообщений
# - группируем по словам,
# - агрегируем с подсчётом количества сообщений для каждого слова,
# переименовываем post_id по смыслу: "частота встречаемости слова",
# - сортируем частоты по убыванию:
df_norm_words_occurrence = (
    df_norm_words
    .groupby(["words"], as_index=False)
    .agg('count')
    .rename(columns={"msg_id": "word_occurrence_freq"})
    .sort_values(by="word_occurrence_freq", ascending=False)
)

df_norm_words_occurrence.head(25)
```

Out[33]:

	words	word_occurrence_freq
478	быть	866
2805	медиафайл	647
2964	мочь	389
1584	есть	374
561	весь	362
3361	нет	300
3210	наш	271
6001	снт	253
3052	надо	252
7378	этот	237
6938	участок	237
1586	ещё	237
6556	только	234
1472	дорога	223
1402	добрый	222
6766	уже	215
758	вопрос	210
1129	год	207
6577	тот	204
6090	сообщение	204
2186	какой	203
6430	такой	202
1322	день	200
1262	данный	191
6435	там	181

In [34]:

```
# Создадим DF с разбивкой по индексам сообщений и указанием встречаемости каждого слова в общем массиве слов
# - объединим DFs с ID публикаций и встречаемостью слов по полю "слова"
# - отсортируем по ID публикаций и встречаемости слов в убывающем порядке
df_norm_words_occurrence_by_msg = (
    df_norm_words.merge(df_norm_words_occurrence, on="words", how='left')
        .sort_values(by=["msg_id", "word_occurrence_freq"], ascending=False)
)
df_norm_words_occurrence_by_msg.sample(7)
```

Out[34]:

	msg_id	words	word_occurrence_freq
14154	2175	быстрый	4
13030	2011	член	58
6381	959	нет	300
37338	5717	устав	29
5587	905	иметься	8
38826	5867	камера	52
36522	5645	территория	81

3.1.1.1 Распределение слов по частотным характеристикам

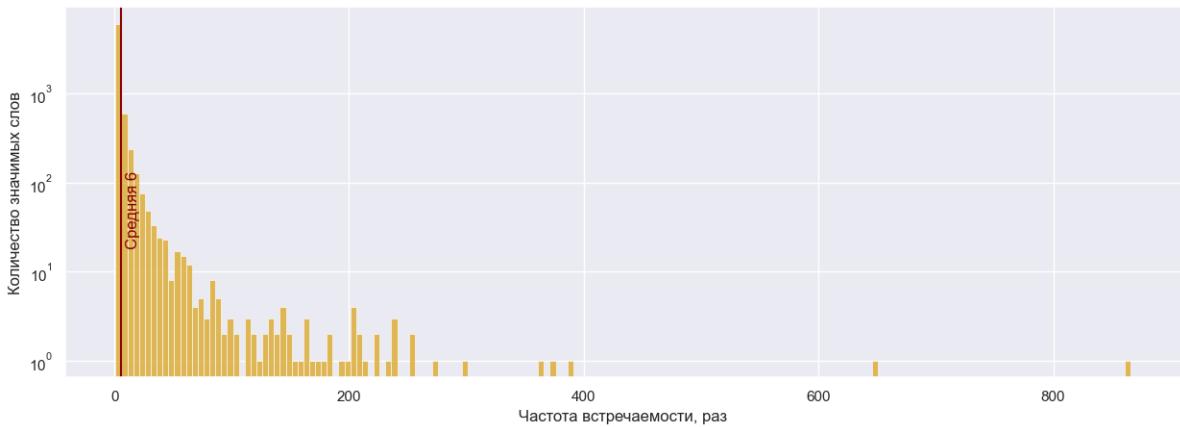
In [35]:

```
# Построим гистограмму распределения количества значимых слов по встречаемости в сообщениях
fig, ax = plt.subplots(figsize=(15,5), ncols=1, nrows=1)
g = sns.histplot(data=df_norm_words_occurrence,
                  x="word_occurrence_freq",
                  binwidth=5,
                  color='goldenrod',
                  ax=ax)

# Mean Line
mean_value = df_norm_words_occurrence.word_occurrence_freq.mean()
dummy = ax.axvline(x=mean_value, color="darkred")
# mean line annotation
dummy = ax.annotate(
    text=f"Средняя {mean_value:.0f}",
    xy=(mean_value+10, 50), # text position
    xycoords='data',
    horizontalalignment='center',
    verticalalignment='center',
    rotation=90, # Rotation
    color='darkred', # Color
    alpha=1,
    fontsize=12,
    fontweight="normal")

dummy = ax.set_yscale('log')
dummy = ax.set_xlabel('Частота встречаемости, раз', fontsize=12)
dummy = ax.set_ylabel('Количество значимых слов', fontsize = 12)
dummy = plt.suptitle(f"Гистограмма распределения встречаемости значимых слов в сообщениях", fontsize=14)
plt.show()
```

Гистограмма распределения встречаемости значимых слов в сообщениях чата



Как видно из графика, у нас есть выбросы по частоте встречаемости. Определим перечень этих слов.

```
In [36]: # calculate word frequencies beyond 3-sigma limit
sigma = df_norm_words_occurrence.word_occurrence_freq.std()
df_norm_words_occurrence[df_norm_words_occurrence.word_occurrence_freq > sigma*3]
```

Out[36]:

	words	word_occurrence_freq
478	быть	866
2805	медиафайл	647
2964	мочь	389
1584	есть	374
561	весь	362
3361	нет	300
3210	наш	271
6001	снт	253
3052	надо	252
7378	этот	237
6938	участок	237
1586	ещё	237
6556	только	234
1472	дорога	223
1402	добрый	222
6766	уже	215
758	вопрос	210
1129	год	207
6577	тот	204
6090	сообщение	204
2186	какой	203
6430	такой	202
1322	день	200
1262	данный	191
6435	там	181
2912	можно	181
6798	улица	180
6736	удалить	173
6109	сосед	166
1448	дом	164
7165	человек	163
2458	который	161
6020	собрание	158
5765	свой	153
1064	где	150
6008	собака	148

	words	word_occurrence_freq
3606	один	145
4765	правление	143
5796	сейчас	143
7085	хороший	142
3960	очень	140
4823	председатель	137
5295	раз	135
5678	сам	133
1445	должный	132
1955	знать	130
5739	свет	126
6170	СПК	125
3415	новый	119
1310	делать	117
7158	чат	113
1221	группа	112
5785	сделать	111
4485	пока	105
5027	проблема	101
5862	сказать	98
400	большой	97
2790	машина	97
7090	хотеть	92
1271	дать	91
1323	деньга	90
1522	другой	89
3445	нужно	89
6669	тут	89
5498	ребёнок	87
3554	общий	84
754	вообще	84
4206	писать	83
1661	забор	82
5982	снег	82
614	видеть	81
2914	мой	81

	words	word_occurrence_freq
6506	территория	81
3446	нужный	80
4271	победа	78
4736	почему	76
2180	каждый	75
1928	здесь	75
2630	линия	74
5270	работать	73
5269	работа	71
1630	жить	69
5790	сегодня	68
2108	информация	67

```
In [37]: vocabulary = (
    df_norm_words_occurrence
    .words
    .count()
)
frequent_words = (
    df_norm_words_occurrence
    .word_occurrence_freq[df_norm_words_occurrence.word_occurrence_freq <= 20]
    .count()
)
most_freq_words = (
    df_norm_words_occurrence
    .word_occurrence_freq[df_norm_words_occurrence.word_occurrence_freq > 20]
    .count()
)

print("Объем словаря значимых слов: ", vocabulary,
      "\n20 и менее раз используется: ", frequent_words,
      "\nБолее 20 раз используется: ", most_freq_words,
      f'\n50% Значимых слов встречаются не более {df_norm_words_occurrence.word_oc
```

Объем словаря значимых слов: 7419
20 и менее раз используется: 7079
Более 20 раз используется: 340
50% Значимых слов встречаются не более 1.0 раз(-а)

Таки образом

1. Общий словарь значимых слов, использованный в сообщениях, составляет 7419 слов. Это достаточно много для такого чата.
2. Подавляющее число значимых слов (более 7070) встречается публикациях менее 20 раз.
3. 340 слов используется менее 20 раз.
4. 50% слов используется не более 1 раза
5. Есть слова, используемые очень часто: от 300 до 866 раз.

Примечательно, что наиболее употребимым является слово "быть", не имеющее специфического значения. Оно встречается 866 раз. Так же выделяется слово "медиафайл" - это сообщения содержащие только изображения, аудио или видео. Всего таких медиафайлов 647. По списку перечню слов с очень высокой частотой встречаемости можно будет судить о наиболее актуальных темах для участников чата. Сделаем это несколько позже.

3.3.3.2. Характеристика сообщений по средней частоте встречаемости используемых слов

Для определения, в какой мере та или иная публикация использует часто встречаемые слова, введём усреднённый показатель: средняя встречаемость слов в сообщении. Для его получения сумму частот встречаемости каждого значимого слова в сообщении разделим на количество значимых слов в этом сообщении.

```
In [38]: # Создадим отдельный DF для данного подраздела.  
# Для этого группируем df_norm_words_occurrence_by_msg по msg_id, подсчитываем в них  
  
# Вычисляем сумму встречаемости слов.  
data3332 = df_norm_words_occurrence_by_msg.groupby(["msg_id"], as_index=False)\  
.agg({"words":'count', "word_occurrence_freq":'sum'})  
  
# Добавляем новый столбец для средней встречаемости слов в сообщении  
data3332["avg_word_occurrence"] = data3332.word_occurrence_freq / data3332.words  
  
# Переименовываем столбцы по смыслу  
data3332.rename(columns={"words":"words_count", "word_occurrence_freq":"total_word_occurrence"}, inplace=True)  
  
# Сохраним индекс df_chat  
df_chat_idx = df_chat.index  
  
# Добавим данные о средней встречаемости слов в публикации в df_chat  
df_chat = df_chat.merge(data3332, on="msg_id", how='left')  
# Восстановим индекс в df_chat  
df_chat.set_index(df_chat_idx, inplace=True)  
  
data3332.sample(3)  
df_chat.sample(3)
```

```
Out[38]:   msg_id  words_count  total_word_occurrence  avg_word_occurrence  
3771      4242           1                      2            2.000000  
471       548           33                     1326          40.181818  
2505      2831           2                      49            24.500000
```

Out[38]:

	user	message	action	day_of_week	DOW_name	hour	msg_id	words_count	t
date_time									
2023-04-02 10:20:00	929 552-10-20 +7	Что устанавливают, кто?		NaN		6	воскресенье	10	5843 1.0
2022-05-25 16:03:00	915 134-67-88 +7	Добрый день! Подскажите, пожалуйста, тариф на ...		NaN		2	среда	16	3255 9.0
2022-03-11 18:15:00	977 839-40-24 +7	У нас к- нект минимальный тариф 800р , один в...		NaN		4	пятница	18	3032 15.0

In [39]:

```
# Построим гистограмму распределения сообщений по средней встречаемости слов
fig, ax = plt.subplots(figsize=(15,5), ncols=1, nrows=1)
g=sns.histplot(
    data=data3332,
    x="avg_word_occurrence",
    binwidth=5,
    kde=True,
    color='firebrick',
    ax=ax)

dummy = ax.set_xlabel('Средняя встречаемость слов в одном сообщении, раз', fontsize=12)
dummy = ax.set_ylabel('Количество сообщений', fontsize = 12)

# Прочертим линию средней
mean_value = data3332.avg_word_occurrence.mean()
dummy = ax.axvline(x=mean_value, color="forestgreen")
# Создание подписи
dummy = ax.annotate(
    text=f"средняя {mean_value:.0f}",
    xy=(mean_value+10, 400), # Положение подписи (чтобы не перекрывать точки графика)
    xycoords='data',
    horizontalalignment='center', # Выравнивание текста по горизонтали
    verticalalignment='center', # Выравнивание текста по вертикали
    rotation=90, # Поворот подписи
    color='forestgreen', # Цвет надписи
    alpha=1,
    fontsize=10,
    fontweight="normal")

# Прочертим линию медианы
median_value = data3332.avg_word_occurrence.median()
dummy = ax.axvline(x=median_value, color="navy")
# Создание подписи
dummy = ax.annotate(
    text=f"медиана {median_value:.0f}",
    xy=(median_value+10, 400), # Положение подписи (чтобы не перекрывать точки графика)
    xycoords='data',
    horizontalalignment='center', # Выравнивание текста по горизонтали
    verticalalignment='center', # Выравнивание текста по вертикали
    rotation=90, # Поворот подписи
    color='navy', # Цвет надписи
```

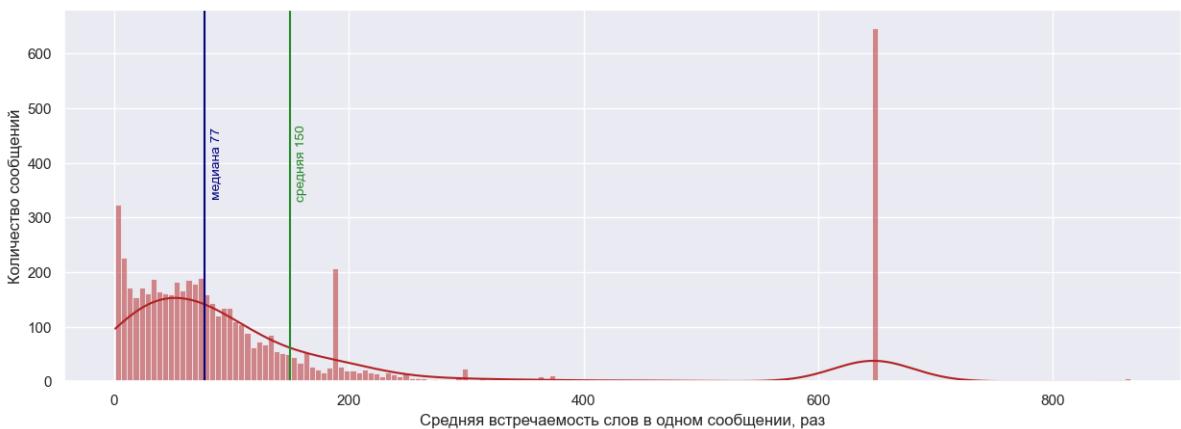
```

alpha=1,
fontsize=10,
fontweight="normal")

dummy = plt.suptitle('Гистограмма распределения сообщений в чате по средней встречаемости слов',
                     fontsize=14)
plt.show()

```

Гистограмма распределения сообщений в чате по средней встречаемости слов



Мы получили полимодальное распределение с очень большим выбросом и с ярко выраженной модой на уровне средней встречаемости около 630 раз. Интересно, что таких сообщений приблизительно такое же количество. Очевидно, что сообщения со средней встречаемостью слов выше 600 - это наиболее короткие сообщения со словом "быть" или содержащие только вложенные медиафайлы.

3.2. Анализ смысловой окрашенности сообщений

Существует целый ряд слов, которые носят позитивную или негативную смысловую окраску. При соответствующих знаниях в области прикладной психологии (например, neuro-linguistic programming - применительно к коммуникации), можно выявить сильные мотивирующие или демотивирующие тексты. Интересно будет выявить, в какой мере слова с различной смысловой окраской используются в сообщениях в чате.

ПРИМЕЧАНИЕ Словари негативных и позитивных слов, которые удалось найти автору, разли чаются по своему объёму: словарь позитивных слов намного шире, словаря негативных слов. Поэтому определение степени негативности текстов может быть искажено в сторону улучшения результата.

3.2.1. Создание перечней позитивно и негативно окрашенных слов

Изучим состав сообщений с точки зрения позитивных (положительных, мотивирующих) слов. Для этого загрузим достаточно большой перечень таких слов, найденный в Интернете на сайте *Positive Words Research*
<https://positivewordsresearch.com/positive-words-russian/>

```
In [40]: # Функция чтения текста по URL адресу (использует библиотеку bs4 BeautifulSoup)
def get_text(url):
    rs = requests.get(url)
    root = BeautifulSoup(rs.content, 'html.parser')
    article = root.select_one('article')
    return article.text

url = "https://everychild.ru/reyting/horoshie-slova-v-russkom-yazyke-primerly/"
positive_words_text = get_text(url)
#print(positive_words_text)
```

Полученный текст необходимо очистить от вступления, заключения и повторяющихся вставок-алфавитных разделителей, а также нормализовать.

ПРИМЕЧАНИЕ

Приведенный ниже код не универсальный, а создан именно для данного текста

```
In [41]: # регулярные выражения для удаления частей текста.
list_regex = [r'\n+', # 1 и более символов перевода строки
              r'(.*)(?=абсолютно)', # любые символы, кроме перевода строки, до слова 'абсолютно'
              r'(?<=ясный)(.*', # любые символы, кроме перевода строки, после слова 'ясный'
              r'положительные слова, которые начинаются с .?', # любой символ (. - точка)
              r'положительные слова, которые начинаются .?'] # любой символ (одна точка)
# вычленить слова

for regex in list_regex:
    positive_words_text = re.sub(regex, ' ', positive_words_text) # выполняем замену
#print(positive_words_text)

# Вычленим слова, очистим от служебных частей речи, нормализуем (аналогично функции Lemmatize)
regex = re.compile(r'\w+') # вычленить слова

list_positive_words = regex.findall(positive_words_text)
# нормализуем список:
# грамемы русского языка: для исключения
# междометия, частицы, союзы, предлоги, личные местоимения
functors_pos = {'INTJ', 'PRCL', 'CONJ', 'PREP', 'NPRO'}
# проверяем и нормализуем слова
list_norm_clean_positive_words = [] # список нормализованных слов в нижнем регистре
for word in list_positive_words:
    # если результат вызова функции не является служебной частью речи или именным существительным
    if pos(word) not in functors_pos:
        # приводим слово к исходной форме (normalized)
        parsed_word = morph.parse(word) # результат вызова morph - список из разных форм
        normalized_word = parsed_word[0].normal_form # выбираем исходную форму слова
        # добавляем исходную форму проверенного слова в список исходных форм
        list_norm_clean_positive_words.append(normalized_word) # добавляем проверенное слово

#print(norm_clean_positive_words_list)
```

```
In [42]: # Сохраним список положительных слов в формате joblib
joblib.dump(list_norm_clean_positive_words, "data/list_positive_words.joblib")
list_norm_clean_positive_words[:25]
```

Out[42]: ['data/list_positive_words.joblib']

```
Out[42]: ['абсолютно',
'адаптироваться',
'аккуратность',
'аккуратный',
'активизироваться',
'активный',
'альtruистический',
'ангельский',
'auténtичный',
'баланс',
'безмятежность',
'безмятежный',
'безопасный',
'бесконечность',
'бесконечный',
'беспристрастный',
'бесстрашный',
'бесценный',
'благодарный',
'радость',
'благодарность',
'благодарный',
'благоприятный',
'благородный',
'благосклонный']
```

Сопоставим каждое слово в df_norm_words_count_by_post и df_norm_words_occurrence со списком нормализованных позитивных слов list_norm_clean_positive_words.

```
In [43]: df_norm_words_occurrence_by_msg = (
    df_norm_words_occurrence_by_msg
    .assign(
        positive = (df_norm_words_occurrence_by_msg
                    .words
                    .apply(lambda x: x in list_norm_clean_positive_words))
    )
)
df_norm_words_occurrence = (
    df_norm_words_occurrence
    .assign(positive = (df_norm_words_occurrence
                        .words
                        .apply(lambda x: x in list_norm_clean_positive_words)))
)
```

```
In [44]: df_norm_words_occurrence_by_msg[df_norm_words_occurrence_by_msg.positive == True].
```

```
Out[44]:   msg_id  words  word_occurrence_freq  positive
2420      468  сейчас            143      True
34387     5372  сейчас            143      True
21620     3214  любовь              8      True
```

```
In [45]: df_norm_words_occurrence[df_norm_words_occurrence.positive == True].sample(3)
```

Out[45]:

	words	word_occurrence_freq	positive
6686	убедительный	1	True
1933	зелёный	4	True
4970	принятие	4	True

Выясним, какие из встречающихся слов в публикациях Skillbox в VK носят негативный (отрицательный, демотивирующий) характер. Для этого загрузим перечень таких слов, найденный в Интернете на сайте [wordsonline.ru](https://wordsonline.ru/samples/negative.html)

<https://wordsonline.ru/samples/negative.html>

ПРИМЕЧАНИЕ

1. Список слов сохранён в файл MS Excel.
2. Слова уже нормализованы.

In [46]:

```
# Создадим список негативных слов, прочитав его из файла Excel
list_negative_words = pd.read_excel("data/negative_words.xlsx", header=0, index_col=0)
#list_negative_words
```

Сопоставим каждое слово в df_norm_words_occurrence_by_post и в df_norm_words_occurrence со списком нормализованных негативных слов negative_word_list.

In [47]:

```
df_norm_words_occurrence_by_msg = (
    df_norm_words_occurrence_by_msg
    .assign(
        negative = (df_norm_words_occurrence_by_msg
                    .words
                    .apply(lambda x: x in list_negative_words))
    )
)
df_norm_words_occurrence = (
    df_norm_words_occurrence
    .assign(
        negative = (
            df_norm_words_occurrence
            .words
            .apply(lambda x: x in list_negative_words))
    )
)
```

In [48]:

```
df_norm_words_occurrence_by_msg[df_norm_words_occurrence_by_msg.negative == True].
```

Out[48]:

	msg_id	words	word_occurrence_freq	positive	negative
13029	2011	обязанность	5	False	True
11932	1821	кражा	4	False	True
21223	3117	беспокойство	2	False	True

In [49]:

```
df_norm_words_occurrence[df_norm_words_occurrence.negative == True].sample(3)
```

Out[49]:

	words	word_occurrence_freq	positive	negative
1151	горе	1	False	True
733	война	9	False	True
3236	негатив	5	False	True

3.2.2. Распределение позитивных и негативных слов по частотным характеристикам

In [50]:

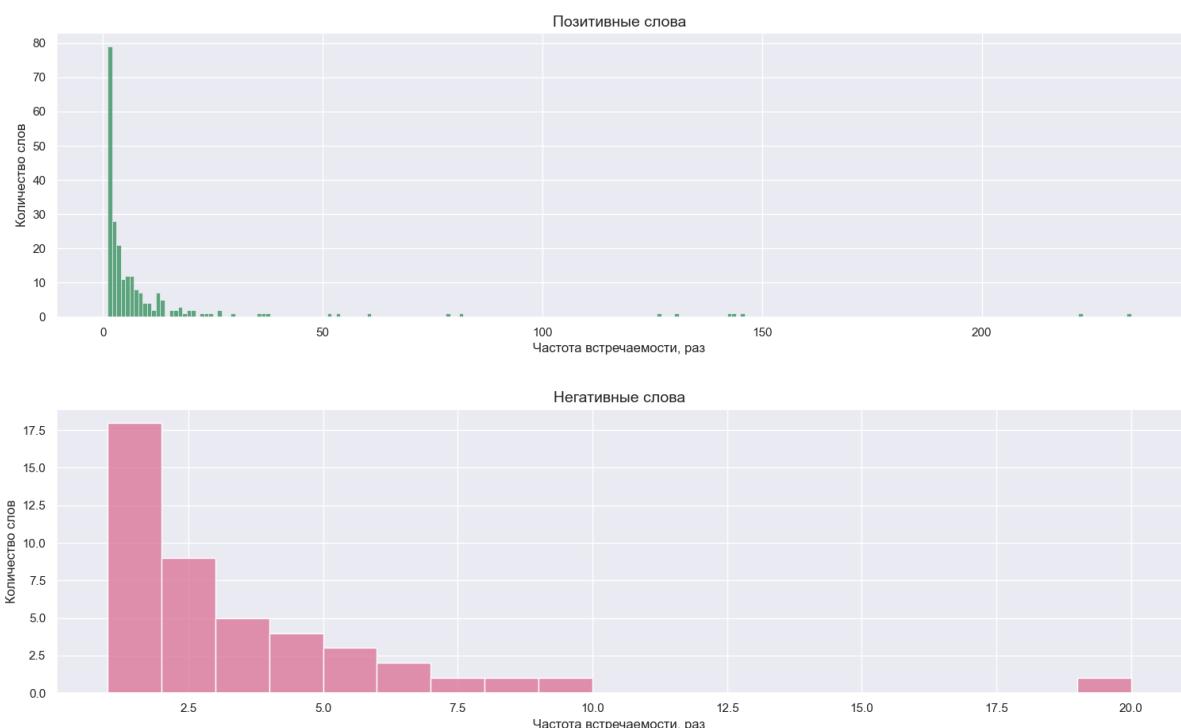
```
# Построим гистограмму распределения количества позитивных и негативных слов по всем
fig, ax = plt.subplots(figsize=(15,9), ncols=1, nrows=2)
fig.tight_layout(h_pad=5)
data1 = df_norm_words_occurrence[df_norm_words_occurrence.positive==True]
data2 = df_norm_words_occurrence[df_norm_words_occurrence.negative==True]
param_list=[[data1, 'seagreen', "Позитивные слова"], [data2,'palevioletred', "Негативные слова"]]

for i, param in enumerate(param_list):
    g = sns.histplot(
        data=param[0],
        x="word_occurrence_freq",
        binwidth=1,
        color=param[1],
        ax=ax[i])

    dummy = ax[i].set_xlabel('Частота встречаемости, раз', fontsize=12)
    dummy = ax[i].set_ylabel('Количество слов', fontsize = 12)
    dummy = ax[i].set_title(f'{param[2]}', fontsize = 14)

dummy = plt.suptitle(f"Гистограмма распределения количества позитивных и негативных слов\nпо встречаемости в сообщениях в чате",
                     fontsize=14, y=1.125)
plt.show()
```

Гистограмма распределения количества позитивных и негативных слов
по встречаемости в сообщениях в чате



```
In [51]: vocabulary = df_norm_words_occurrence.words.count()
positive_voc = df_norm_words_occurrence.positive.sum()
negative_voc = df_norm_words_occurrence.negative.sum()
print(f"Общее количество позитивных слов в словарном запасе: {positive_voc}, "
      f"то есть {(positive_voc/vocabulary*100):.2f}%")
print(f"Общее количество негативных слов в словарном запасе: {negative_voc}, "
      f"то есть {(negative_voc/vocabulary*100):.2f}%)")
```

Общее количество позитивных слов в словарном запасе: 233, то есть 3.14%
 Общее количество негативных слов в словарном запасе: 45, то есть 0.61%

3.2.3. Характеристика сообщений по степени позитивности и негативности

Проанализируем соотношение позитивных и негативных слов к общему числу слов в каждой отдельной публикации. В отличие от средней встречаемости слов в публикации, которая показывала, насколько та или иная публикация использует часто встречающиеся слова, здесь мы выясним общую степень "позитивности" и "негативности" каждой публикаций.

```
In [52]: # Создадим отдельный DF для данного подраздела.
# Для этого:

# - в новом DF группируем df_norm_words_occurrence_by_msg по номерам сообщений,
# - подсчитываем в них значимые позитивные и негативные слова и
# - вычисляем отношения количества позитивных и негативных слов к общему числу слов
df_323=df_norm_words_occurrence_by_msg.drop(columns="word_occurrence_freq")
df_323=df_norm_words_occurrence_by_msg.groupby(["msg_id"])\n
.agg({"words":'count', "positive":'sum', "negative":'sum'})

# Добавляем новые столбцы для доли позитивных и негативных слов в общем количестве
df_323["pos_fraction"] = df_323.positive / df_323.words
df_323["neg_fraction"] = df_323.negative / df_323.words

# Переименовываем столбцы по смыслу
df_323.rename(columns={"words":"words_count", "word_occurrence_freq":"total_word_oc",
                      "positive":"pos_word_count", "negative":"neg_word_count"}, inplace=True)
df_323.sample(5)
```

Out[52]:

	words_count	pos_word_count	neg_word_count	pos_fraction	neg_fraction
msg_id					
3619	26	1	1	0.038462	0.038462
271	8	0	0	0.000000	0.000000
5657	1	0	0	0.000000	0.000000
5091	7	1	0	0.142857	0.000000
6275	1	0	0	0.000000	0.000000

```
In [53]: # Добавим данные о доле позитивных и негативных слов в сообщениях в df_chat
df_chat = df_chat.merge(df_323[["pos_fraction", "neg_fraction" ]], on="msg_id", how="left")
# Восстановим индекс в df_chat
df_chat.set_index(df_chat_idx, inplace=True)
df_chat.sample(1)
```

Out[53]:

	user	message	action	day_of_week	DOW_name	hour	msg_id	words_count	text
date_time									
2022-08-08 15:38:00	+7 915 449-73-74	Если короткое замыкание было после 74	электроощита...	NaN	0	понедельник	15	3645	35.0

In [54]:

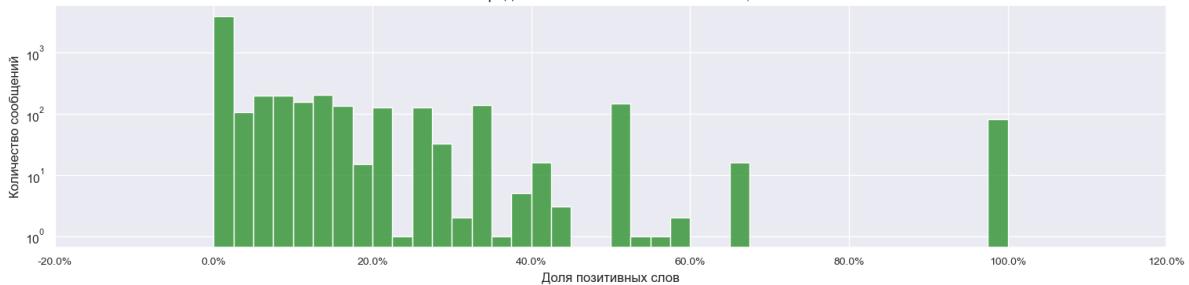
```
fig, ax = plt.subplots(figsize=(15,8), ncols=1, nrows=2)
fig.tight_layout(h_pad=5)
data = df_323
list_param=[["pos_fraction", 'forestgreen', "позитивности", "позитивных"],
            ["neg_fraction",'firebrick', "негативности", "негативных"]]

for i, param in enumerate(list_param):
    g = sns.histplot(
        data=data,
        x=param[0],
        binwidth=0.025,
        color=param[1],
        ax=ax[i])

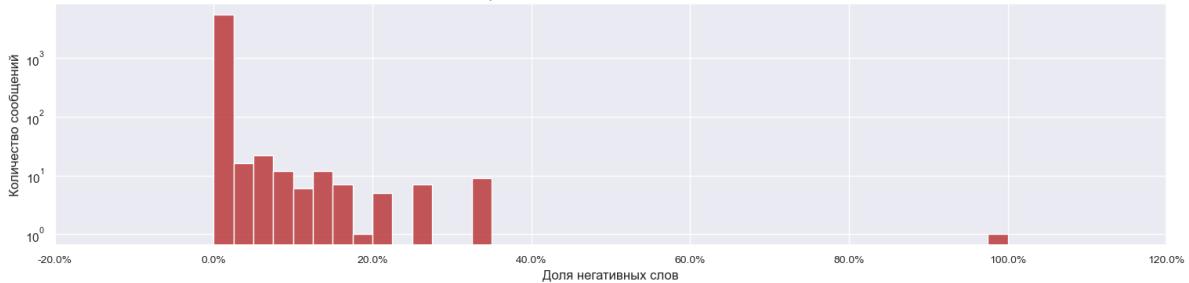
    dummy = ax[i].set_xlabel(f"Доля {param[3]} слов", fontsize=12)
    dummy = ax[i].set_ylabel('Количество сообщений', fontsize = 12)
    dummy = ax[i].set_title(f"Распределение по степени {param[2]}, %", fontsize = 12)
    # изменим подписи шкалы Y, определим их в процентах
    # a) исправляем User Warning: FixedFormatter should only be used together with
    ticks_loc = ax[i].get_xticks().tolist()
    dummy = ax[i].set_xticks(ticks_loc)
    # б) устанавливаем формат и значения
    xlabelss = [f"{{y * 100}}:{.1f}%" for y in ticks_loc]
    dummy = ax[i].set_xticklabels(xlabelss, fontsize=10)
    dummy = ax[i].set_yscale('log')
dummy = plt.suptitle(f"Гистограммы распределения сообщений в чате по смысловой окраске", fontsize=14, y=1.0625)
plt.show()
```

Гистограммы распределения сообщений в чате по смысловой окраске

Распределение по степени позитивности, %



Распределение по степени негативности, %



```
In [55]: texts_num=df_chat.msg_id.count()
texts_pos = df_323.pos_fraction[df_323.pos_fraction != 0].count()
texts_neg = df_323.neg_fraction[df_323.neg_fraction != 0].count()
print(f"Общее количество сообщений {texts_num} \n"
      f"Общее количество сообщений, содержащих позитивные слова {texts_pos}, "
      f"то есть {(texts_pos/texts_num*100):.1f}%\n"
      f"Общее количество сообщений, содержащих негативные слова {texts_neg}, "
      f"то есть {(texts_neg/texts_num*100):.1f}%", )
```

Общее количество сообщений 6301

Общее количество сообщений, содержащих позитивные слова 1734, то есть 27.5%
Общее количество сообщений, содержащих негативные слова 110, то есть 1.7%

3.2.4. Группировка участников по степени смысловой окрашенности сообщений

ПРИМЕЧАНИЕ

Группировка по позитивной и по негативной смысловой окрашенности сообщений производятся раздельно и независимо друг от друга, поэтому участники могут упоминаться в обеих группировках одновременно.

```
In [56]: df_324 = (
    df_chat[['user', "pos_fraction", "neg_fraction"]]
    .dropna(subset=["pos_fraction", "neg_fraction"])
    .groupby('user', as_index=False)
    .agg('mean')
)
```

```
In [57]: print("15 участников с максимальной средней ПОЗИТИВНОЙ смысловой окраской сообщений")
(
    df_324[['user', 'pos_fraction']]
    .sort_values(by=['pos_fraction'], ascending=False)
    .nlargest(15, 'pos_fraction')
)
```

15 участников с максимальной средней ПОЗИТИВНОЙ смысловой окраской сообщений

Out[57]:

	user	pos_fraction
16	+7 905 709-53-83	0.458874
86	+7 964 780-02-27	0.373568
2	+7 903 170-62-90	0.333333
98	+7 977 839-40-24	0.304762
63	+7 925 154-61-34	0.175967
46	+7 916 675-72-39	0.145933
24	+7 915 134-67-88	0.138292
37	+7 916 500-70-07	0.137311
70	+7 926 248-32-75	0.133542
88	+7 965 280-67-40	0.130899
77	+7 926 582-05-88	0.130379
62	+7 925 126-40-76	0.125000
100	+7 977 937-32-70	0.123302
18	+7 905 746-33-57	0.123061
68	+7 926 180-22-18	0.118997

In [58]:

```
print("15 участников с максимальной средней НЕГАТИВНОЙ смысловой окраской сообщений")
(
    df_324[['user', 'neg_fraction']]
    .sort_values(by=['neg_fraction'], ascending=False)
    .nlargest(15, 'neg_fraction')
)
```

15 участников с максимальной средней НЕГАТИВНОЙ смысловой окраской сообщений

Out[58]:

		user	neg_fraction
84	+7 962 999-94-04		0.016667
96	+7 977 606-55-92		0.013333
5	+7 903 241-45-76		0.011905
30	+7 916 149-49-26		0.010332
79	+7 926 667-77-23		0.008977
89	+7 966 004-05-06		0.007576
7	+7 903 578-91-95		0.006834
77	+7 926 582-05-88		0.006536
60	+7 925 066-47-95		0.005881
68	+7 926 180-22-18		0.005652
32	+7 916 211-85-14		0.005556
94	+7 977 449-73-06		0.004975
29	+7 916 140-17-25		0.004444
75	+7 926 565-55-49		0.003623
105	+7 985 444-72-00		0.003235

4. Анализ актуальных тем

4.1. Составление словаря актуальных тем и ключевых слов.

In [59]:

```
# dictionary of important topics
list_topics = ['выход_из_СНТ', 'дети_поведение', 'дороги_состояние', 'животные_содержание',
               'мусор', 'наркоманы', 'территория_СНТ_режим', 'транспорт_движение', 'ЖКУ']

dict_topics = {'выход_из_СНТ': ['город', 'ижс', 'луч', 'москва', 'посёлок'],
               'дети_поведение': ['бегать', 'велосипед', 'гонять', 'гулять', 'дети',
                                  'самокат', 'скейтер'],
               'дороги_состояние': ['асфальт', 'гололед', 'гравий', 'грязь', 'дорога',
                                     'ливневка', 'ливнёвка', 'обочина', 'подсыпать',
                                     'пыль', 'разъезд', 'снег', 'сугроб', 'щебенка',
                                     'щебёнка', 'яма'],
               'животные_содержание': ['бездомный', 'выгул', 'животное', 'кот', 'кошка',
                                         'намордник', 'напасть', 'непривитый', 'ничей',
                                         'прививать', 'прививка', 'привитый', 'собака',
                                         'стерилизация', 'стерилизовать', 'укус', 'укусить'],
               'ЖКУ': ['вода', 'водопровод', 'газ', 'газопровод', 'свет', 'освещение',
                       'обслуживание', 'электричество', 'электроэнергия'],
               'интернет': ['интернет', 'передача', 'провайдер', 'соединение'],
               'мусор': ['банка', 'вонь', 'вонять', 'выбрасывать', 'выбросить', 'выбросить',
                         'вынести', 'выносить', 'крыса', 'мусор', 'пакет', 'пластик',
                         'раздельный', 'стекло'],
               'наркоманы': ['закладка', 'наркоман', 'наркотик'],
               'территория_СНТ_режим': ['видеокамера', 'видеонаблюдение', 'ворота',
                                         'калитка', 'камера', 'наблюдение', 'охрана',
                                         'проходить', 'сторож', 'территория', 'шлагбаум']}
```

```

    'транспорт_движение': ['автомобиль', 'большегрузный', 'грузовой', 'парковать', 'парковка', 'скоростной', 'скорость', 'проноситься', 'пронестись'],
    'управление_СНТ': ['повестка', 'правление', 'председатель', 'протокол', 'решение', 'решить', 'собрание', 'устав'],
    'финансы_СНТ': ['аудит', 'аудиторский', 'бухгалтерия', 'бухгалтерский', 'задолженность', 'зарплата', 'оплата', 'плата', 'ставка', 'сумма', 'тариф', 'цена']}

```

In [60]:

```

# List of keywords
list_keywords = []
for key, item in dict_topics.items():
    list_keywords.extend(item)
list_keywords.sort()
print(list_keywords)

```

```

['автомобиль', 'асфальт', 'аудит', 'аудиторский', 'банка', 'бегать', 'бездомный', 'большегрузный', 'бухгалтерия', 'бухгалтерский', 'быстро', 'бюджет', 'велосипед', 'взнос', 'видеокамера', 'видеонаблюдение', 'вода', 'водопровод', 'вонь', 'вонять', 'ворота', 'въезд', 'выбрасывать', 'выбросить', 'вывоз', 'выгул', 'выкидывать', 'выкинуть', 'вынести', 'выносить', 'газ', 'газопровод', 'гололед', 'гонять', 'город', 'гравий', 'грузовой', 'грязь', 'гулять', 'дети', 'долг', 'дорога', 'животное', 'забор', 'задолженность', 'закладка', 'замок', 'зарплата', 'засыпать', 'ижс', 'интернет', 'калитка', 'камера', 'канава', 'квадроцикл', 'кот', 'кошка', 'крыса', 'кусатъ', 'кусаться', 'ливневка', 'ливнёвка', 'луч', 'машина', 'москва', 'мусор', 'наблюдение', 'намордник', 'напасть', 'наркоман', 'наркотик', 'непривитый', 'нестись', 'ничейный', 'обочина', 'обслуживание', 'оплата', 'освещение', 'охранник', 'ошейник', 'пакет', 'парковать', 'парковка', 'передача', 'пластиковый', 'плата', 'платить', 'повестка', 'проводок', 'подсыпать', 'покрытие', 'помойка', 'посыпать', 'посёлок', 'правление', 'председатель', 'прививать', 'прививка', 'привитый', 'проводайдер', 'пронестись', 'проноситься', 'протокол', 'проход', 'проходить', 'пыль', 'раздельный', 'разъезд', 'расход', 'ребенок', 'решение', 'решить', 'самокат', 'свет', 'скоростной', 'скорость', 'скутер', 'смета', 'снег', 'собака', 'собрание', 'соединение', 'ставка', 'стекло', 'стерилизация', 'стерилизовать', 'сторож', 'сугроб', 'сумма', 'тариф', 'территория', 'укус', 'укусить', 'устав', 'цена', 'шлагбаум', 'щебенка', 'щебень', 'щебёнка', 'электричество', 'электроэнергия', 'яма']

```

4.2. Частота упоминаний ключевых слов и актуальных тем в сообщениях

In [61]:

```

# Локальная функция получения ключа словаря

def get_key(elem_):
    """ Локальная функция вывода ключа словаря по нахождению элемента в значении соответствующему данному ключу"""
    for key_, value_ in dict_topics.items():
        if elem_ in value_:
            return key_
    return "Без определённой темы"

```

In [62]:

```

print("Количество упоминаний ключевых слов, в порядке убывания")

(
    df_norm_words_occurrence[
        df_norm_words_occurrence
            .words
            .apply(lambda x: x in list_keywords)]
        .assign(topic=df_norm_words_occurrence
            .words
            .apply(lambda x: get_key(elem_=x)))

```

)

Количество упоминаний ключевых слов, в порядке убывания

Out[62]:

	words	word_occurrence_freq	positive	negative	topic
1472	дорога	223	False	False	дороги_состояние
6020	собрание	158	False	False	управление_CHT
6008	собака	148	False	False	животные_содержание
4765	правление	143	False	False	управление_CHT
4823	председатель	137	False	False	управление_CHT
5739	свет	126	True	False	ЖКУ
2790	машина	97	False	False	транспорт_движение
1661	забор	82	False	False	территория_CHT_режим
5982	снег	82	False	False	дороги_состояние
6506	территория	81	False	False	территория_CHT_режим
687	вода	62	False	False	ЖКУ
5556	решение	60	False	False	управление_CHT
4689	посёлок	59	False	False	выход_из_CHT
767	ворота	59	False	False	территория_CHT_режим
7344	электричество	58	False	False	ЖКУ
862	въезд	55	False	False	территория_CHT_режим
2192	камера	52	False	False	территория_CHT_режим
206	асфальт	51	False	False	дороги_состояние
2950	москва	48	False	False	выход_из_CHT
2466	кошка	47	False	False	животные_содержание
1236	гулять	45	False	False	дети_поведение
2095	интернет	45	False	False	интернет
1155	город	44	False	False	выход_из_CHT
5557	решить	42	False	False	управление_CHT
4237	платить	41	False	False	финансы_CHT
1615	животное	40	False	False	животные_содержание
2982	мусор	40	False	False	мусор
3953	охрана	37	False	False	территория_CHT_режим
1044	газ	36	False	False	ЖКУ
595	взнос	36	False	False	финансы_CHT
260	бегать	31	False	False	дети_поведение
5033	провайдер	31	False	False	интернет
2195	канава	31	False	False	дороги_состояние
6881	устав	29	False	False	управление_CHT
6443	тариф	29	False	False	финансы_CHT
7266	шлагбаум	28	False	False	территория_CHT_режим

	words	word_occurrence_freq	positive	negative	topic
1994	ижс	27	False	False	выход_из_СНТ
2452	кот	27	False	False	животные_содержание
7121	цена	26	False	False	финансы_СНТ
5189	проходить	25	False	False	территория_СНТ_режим
5889	скорость	18	False	False	транспорт_движение
5466	расход	18	False	False	финансы_СНТ
3967	ошейник	18	False	False	животные_содержание
2188	калитка	16	False	False	территория_СНТ_режим
3954	охранник	16	False	False	территория_СНТ_режим
609	видеонаблюдение	16	False	False	территория_СНТ_режим
3736	освещение	15	False	False	ЖКУ
4010	парковка	15	False	False	транспорт_движение
2530	крыса	15	False	False	мусор
474	быстро	15	True	False	транспорт_движение
3686	оплата	14	False	False	финансы_СНТ
6372	сумма	14	False	False	финансы_СНТ
4577	помойка	14	False	False	мусор
2685	луч	13	False	False	выход_из_СНТ
6273	стерилизация	13	False	False	животные_содержание
4295	поводок	13	False	False	животные_содержание
480	бюджет	12	False	False	финансы_СНТ
4891	прививка	12	False	False	животные_содержание
693	водопровод	12	False	False	ЖКУ
888	вывоз	11	False	False	мусор
4291	повестка	10	False	False	управление_СНТ
1146	гонять	10	False	False	дети_поведение
215	аудит	9	False	False	финансы_СНТ
7358	электроэнергия	9	False	False	ЖКУ
3983	пакет	9	False	False	мусор
7407	яма	9	False	False	дороги_состояние
3533	обслуживание	8	False	False	ЖКУ
4502	покрытие	8	False	False	дороги_состояние
929	выкинуть	8	False	False	мусор
1744	закладка	8	False	False	наркоманы
1885	засыпать	8	False	False	дороги_состояние
468	бухгалтерия	8	False	False	финансы_СНТ

	words	word_occurrence_freq	positive	negative	topic
5188	проход	7	False	False	территория_СНТ_режим
1859	зарплата	7	False	False	финансы_СНТ
5963	смета	7	False	False	финансы_СНТ
1785	замок	6	False	False	территория_СНТ_режим
1229	грязь	6	False	True	дороги_состояние
271	бездомный	6	False	False	животные_содержание
2621	ливневка	6	False	False	дороги_состояние
1436	долг	6	False	True	финансы_СНТ
6274	стерилизовать	6	False	False	животные_содержание
6268	стекло	6	False	False	мусор
4424	подсыпать	6	False	False	дороги_состояние
5381	разъезд	6	False	False	дороги_состояние
4071	передача	6	False	False	интернет
3115	напасть	5	False	False	животные_содержание
3017	наблюдение	5	False	False	территория_СНТ_режим
95	автомобиль	5	False	False	транспорт_движение
6792	укусить	5	False	False	животные_содержание
5253	пыль	5	False	False	дороги_состояние
4236	плата	4	False	False	финансы_СНТ
535	велосипед	4	False	False	дети_поведение
1720	задолженность	4	False	False	финансы_СНТ
3105	намордник	4	False	False	животные_содержание
217	аудиторский	4	False	False	финансы_СНТ
3512	обочина	4	False	False	дороги_состояние
6350	сугроб	4	False	False	дороги_состояние
877	выбрасывать	4	False	False	мусор
881	выбросить	4	False	False	мусор
5177	протокол	3	False	False	управление_СНТ
4233	пластиковый	3	False	False	мусор
4008	парковать	3	False	False	транспорт_движение
5339	раздельный	3	False	False	мусор
1216	грузовой	3	False	False	транспорт_движение
947	выносить	3	False	False	мусор
899	выгул	3	False	False	животные_содержание
928	выкидывать	2	False	False	мусор
3359	нестись	2	False	False	транспорт_движение

	words	word_occurrence_freq	positive	negative	topic
5900	скутер	2	False	False	дети_поведение
2551	кусать	2	False	False	животные_содержание
5888	скоростной	2	False	False	транспорт_движение
3144	наркоман	2	False	False	наркоманы
5692	самокат	1	False	False	дети_поведение
7300	щебёнка	1	False	False	дороги_состояние
6791	укус	1	False	False	животные_содержание
6054	соединение	1	True	False	интернет
397	большегрузный	1	False	False	транспорт_движение
945	вынести	1	False	False	мусор
1181	гравий	1	False	False	дороги_состояние
608	видеокамера	1	False	False	территория_CHT_режим
4686	посыпать	1	False	False	дороги_состояние
2552	кусаться	1	False	False	животные_содержание
3407	ничейный	1	False	False	животные_содержание
3329	непривитый	1	False	False	животные_содержание

```
In [63]: print("Количество упоминаний ключевых слов в разрезе актуальных тем, в порядке убывания")

# Выберем из df_norm_words_occurrence (DF с частотами упоминания слов) только слова,
# упомянутые в словаре актуальных тем, определим по соответствующим ключам словаря topics
# по темам, отберём строки, начиная со второй (исключим "Topic doesn't exist"), посчитаем
# количество упоминаний для каждого слова и темы
(
    df_norm_words_occurrence
        .assign(topic=df_norm_words_occurrence
                    .words
                    .apply(lambda x: get_key(elem_=x)))
        )
[[ 'topic', "word_occurrence_freq"]]
    .groupby('topic')
    .agg('sum')
    .iloc[1:,:]
    .sort_values(by='word_occurrence_freq', ascending=False)
)
```

Количество упоминаний ключевых слов в разрезе актуальных тем, в порядке убывания

Out[63]:

word_occurrence_freq

topic	
управление_СНТ	582
территория_СНТ_режим	486
дороги_состояние	452
животные_содержание	353
ЖКУ	326
финансы_СНТ	239
выход_из_СНТ	191
транспорт_движение	161
мусор	123
дети_поведение	93
интернет	83
наркоманы	10

In [64]:

```
# Создадим промежуточный DF из DF с частотами встречаемости слов с указанием ID сообщений
# актуальных тем
df_a = (
    df_norm_words_occurrence_by_msg[['msg_id', 'words', 'word_occurrence_freq']]
    .assign(topic=df_norm_words_occurrence_by_msg
           .words
           .apply(lambda x: get_key(elem_=x)
                 )
           )
)
# Создадим промежуточный DF из основного, взяв только участников и id сообщений
df_b = df_chat[['user', 'msg_id']].reset_index()
```

In [65]:

```
# Объединим 2 промежуточных DF по общему столбцу ID сообщений.
df_total = df_a.merge(df_b, on='msg_id', how='left')
df_total.sample(5)
```

Out[65]:

	msg_id	words	word_occurrence_freq	topic	date_time	user
26195	2543	собака	148	животные_содержание	2021-12-25 12:19:00	+7 920 213- 01-91
741	6176	сообщение	204	Без определённой темы	2023-04-07 17:50:00	+7 985 769- 03-22
7663	5474	дом	164	Без определённой темы	2023-03-30 19:01:00	+7 903 578- 91-95
6354	5648	нужда	4	Без определённой темы	2023-03-31 14:31:00	+7 929 552- 10-20
19219	3643	открытый	12	Без определённой темы	2022-08-08 15:29:00	+7 915 449- 73-74

In [66]:

```

print("Количество упоминаний ключевых слов в разрезе участников")
print("в порядке убывания")
print("(Наиболее заинтересованные участники)")
(
    df_total[
        df_total
        .words
        .apply(lambda x: x in list_keywords)]
    .groupby('user', as_index=False)
    .agg('count')[['user', 'words']]
    ]
    .sort_values(by='words', ascending=False)
    .rename(columns={"words": "words_count"})
)

```

Количество упоминаний ключевых слов в разрезе участников
 в порядке убывания
 (Наиболее заинтересованные участники)

Out[66]:

		user	words_count
84	+7 985 769-03-22		628
80	+7 985 273-58-42		291
81	+7 985 444-72-00		184
11	+7 905 500-38-87		170
44	+7 925 066-47-95		124
19	+7 915 449-73-74		111
23	+7 916 149-49-26		99
61	+7 929 552-10-20		84
5	+7 903 578-91-95		82
71	+7 977 472-96-06		67
47	+7 926 088-22-70		63
50	+7 926 226-65-66		56
51	+7 926 248-32-75		52
18	+7 915 134-67-88		49
49	+7 926 180-22-18		48
59	+7 926 667-77-23		47
2	+7 903 222-18-60		44
82	+7 985 456-07-35		41
55	+7 926 565-55-49		37
27	+7 916 300-00-18		35
69	+7 977 322-86-98		35
48	+7 926 112-00-22		34
66	+7 966 004-05-06		34
85	+7 991 140-60-88		32
64	+7 965 238-36-37		30
29	+7 916 600-01-65		29
77	+7 985 222-38-78		28
32	+7 916 601-52-22		26
21	+7 915 496-11-38		24
65	+7 965 280-67-40		23
52	+7 926 278-42-47		21
76	+7 977 937-32-70		21
22	+7 916 140-17-25		20
16	+7 915 002-53-33		19
3	+7 903 241-45-76		19
28	+7 916 500-70-07		18

user	words_count
17 +7 915 002-59-99	18
42 +7 920 213-01-91	17
63 +7 964 649-09-13	16
56 +7 926 581-52-04	16
37 +7 916 683-85-99	16
57 +7 926 582-05-88	14
33 +7 916 606-95-93	13
41 +7 916 978-27-70	12
15 +7 905 748-63-90	12
14 +7 905 746-33-57	12
38 +7 916 696-85-88	12
25 +7 916 211-85-14	12
8 +7 903 724-66-50	12
6 +7 903 678-64-96	11
36 +7 916 683-27-69	11
53 +7 926 373-74-11	10
43 +7 920 545-19-08	10
31 +7 916 600-93-58	10
46 +7 925 154-61-34	9
10 +7 903 788-37-81	9
4 +7 903 554-54-96	8
75 +7 977 851-20-56	8
70 +7 977 449-73-06	8
68 +7 977 100-47-70	8
13 +7 905 713-22-77	7
72 +7 977 606-55-92	7
24 +7 916 157-49-24	6
58 +7 926 627-58-44	6
34 +7 916 671-97-31	5
62 +7 962 999-94-04	5
35 +7 916 675-72-39	5
40 +7 916 793-06-50	5
39 +7 916 724-18-38	5
73 +7 977 749-94-27	5
45 +7 925 075-81-04	4
26 +7 916 284-73-42	4

	user	words_count
67	+7 968 353-19-99	4
20	+7 915 458-09-76	4
74	+7 977 839-40-24	3
60	+7 926 838-32-38	2
12	+7 905 709-53-83	2
78	+7 985 228-48-38	2
30	+7 916 600-54-16	2
54	+7 926 483-30-24	1
79	+7 985 257-68-29	1
1	+7 903 188-85-86	1
9	+7 903 763-79-96	1
7	+7 903 684-24-14	1
83	+7 985 710-78-10	1
0	+7 903 003-43-23	1

4.3. Количество актуальных тем по каждому заинтересованному участнику

In [67]: `df_total.sample(3)`

	msg_id	words	word_occurrence_freq	topic	date_time	user
20493	3379	идеальный	2	Без определенной темы	2022-06-29 13:23:00	+7 985 273-58- 42
3440	5938	видеть	81	Без определенной темы	2023-04-02 14:51:00	+7 929 552-10- 20
12368	4825	личка	36	Без определенной темы	2023-01-06 18:34:00	+7 977 851-20- 56

In [68]: `print("Количество уникальных актуальных тем в разрезе участников")`

```
# Подсчитаем количество уникальных актуальных тем, приходящихся на одного участника

df43 = (
    df_total[
        df_total
            .topic != "Без определённой темы"
    ]
    .groupby('user', as_index=False)
    .agg('nunique')[['user', 'topic']]
)
.sort_values(by='topic', ascending=False)
.rename(columns={"topic": "unique_topics"})
```

)
df43

Количество уникальных актуальных тем в разрезе участников

Out[68]:

		user	unique_topics
50	+7 926 226-65-66		12
11	+7 905 500-38-87		11
84	+7 985 769-03-22		11
81	+7 985 444-72-00		11
44	+7 925 066-47-95		11
47	+7 926 088-22-70		10
69	+7 977 322-86-98		10
66	+7 966 004-05-06		10
65	+7 965 280-67-40		10
52	+7 926 278-42-47		10
23	+7 916 149-49-26		10
80	+7 985 273-58-42		10
82	+7 985 456-07-35		10
59	+7 926 667-77-23		9
61	+7 929 552-10-20		9
63	+7 964 649-09-13		9
51	+7 926 248-32-75		9
49	+7 926 180-22-18		9
85	+7 991 140-60-88		9
2	+7 903 222-18-60		9
5	+7 903 578-91-95		9
19	+7 915 449-73-74		8
29	+7 916 600-01-65		8
77	+7 985 222-38-78		8
55	+7 926 565-55-49		8
76	+7 977 937-32-70		8
71	+7 977 472-96-06		8
33	+7 916 606-95-93		8
22	+7 916 140-17-25		8
15	+7 905 748-63-90		8
18	+7 915 134-67-88		8
17	+7 915 002-59-99		8
16	+7 915 002-53-33		8
27	+7 916 300-00-18		8
14	+7 905 746-33-57		7
56	+7 926 581-52-04		7

user	unique_topics
21 +7 915 496-11-38	7
31 +7 916 600-93-58	7
57 +7 926 582-05-88	7
8 +7 903 724-66-50	6
38 +7 916 696-85-88	6
72 +7 977 606-55-92	6
32 +7 916 601-52-22	6
75 +7 977 851-20-56	5
53 +7 926 373-74-11	5
62 +7 962 999-94-04	5
64 +7 965 238-36-37	5
70 +7 977 449-73-06	5
43 +7 920 545-19-08	5
48 +7 926 112-00-22	5
46 +7 925 154-61-34	5
3 +7 903 241-45-76	5
36 +7 916 683-27-69	5
37 +7 916 683-85-99	5
10 +7 903 788-37-81	5
28 +7 916 500-70-07	4
25 +7 916 211-85-14	4
13 +7 905 713-22-77	4
73 +7 977 749-94-27	4
40 +7 916 793-06-50	4
41 +7 916 978-27-70	4
6 +7 903 678-64-96	4
42 +7 920 213-01-91	4
26 +7 916 284-73-42	4
24 +7 916 157-49-24	3
68 +7 977 100-47-70	3
4 +7 903 554-54-96	3
67 +7 968 353-19-99	3
45 +7 925 075-81-04	3
39 +7 916 724-18-38	3
30 +7 916 600-54-16	2
35 +7 916 675-72-39	2

	user	unique_topics
20	+7 915 458-09-76	2
60	+7 926 838-32-38	2
74	+7 977 839-40-24	2
12	+7 905 709-53-83	2
34	+7 916 671-97-31	2
54	+7 926 483-30-24	1
83	+7 985 710-78-10	1
1	+7 903 188-85-86	1
9	+7 903 763-79-96	1
7	+7 903 684-24-14	1
79	+7 985 257-68-29	1
78	+7 985 228-48-38	1
58	+7 926 627-58-44	1
0	+7 903 003-43-23	1

4.4. Количество участников по каждой актуальной теме

```
In [69]: print(" Число уникальных активных участников по каждой актуальной теме\n",
           "и их процент от общего числа активных и всех участников")

# Подсчитаем количество участников, обсуждавших ту или иную тему
df44 = (
    df_total[
        df_total
            .topic != "Без определённой темы"
    ]
    .groupby('topic', as_index=False)
    .agg('nunique')[['topic', 'user']]
)
df44
    .sort_values(by='user', ascending=False)
    .rename(columns={"user": "number_of_users"})
)
(
    df44
    .assign(per_cent_of_active_users = df44.number_of_users / n_active_users * 100
           per_cent_of_all_users = df44.number_of_users / n_users * 100)
    .reset_index(drop=True)
)
```

Число уникальных активных участников по каждой актуальной теме
и их процент от общего числа активных и всех участников

Out[69]:

	topic	number_of_users	per_cent_of_active_users	per_cent_of_all_users
0	территория_СНТ_режим	62	63.917526	27.312775
1	ЖКУ	53	54.639175	23.348018
2	дороги_состояние	53	54.639175	23.348018
3	транспорт_движение	50	51.546392	22.026432
4	животные_содержание	47	48.453608	20.704846
5	управление_СНТ	47	48.453608	20.704846
6	финансы_СНТ	45	46.391753	19.823789
7	выход_из_СНТ	44	45.360825	19.383260
8	дети_поведение	36	37.113402	15.859031
9	мусор	33	34.020619	14.537445
10	интернет	31	31.958763	13.656388
11	наркоманы	7	7.216495	3.083700

4.5. Динамика обсуждения актуальных тем

```
In [71]: # Выбираем данные для графика, создаём столбец с периодичностью 1 месяц, группируем
# темам, подсчитываем количество сообщений
data44 = (
    df_total[df_total.topic != "Без определённой темы"][[ 'date_time', 'topic', 'msg_id']
    .assign(year_month = df_total.date_time.dt.to_period("M"))
    .groupby(['year_month', 'topic'], as_index=False)
    .agg('count')
)
```

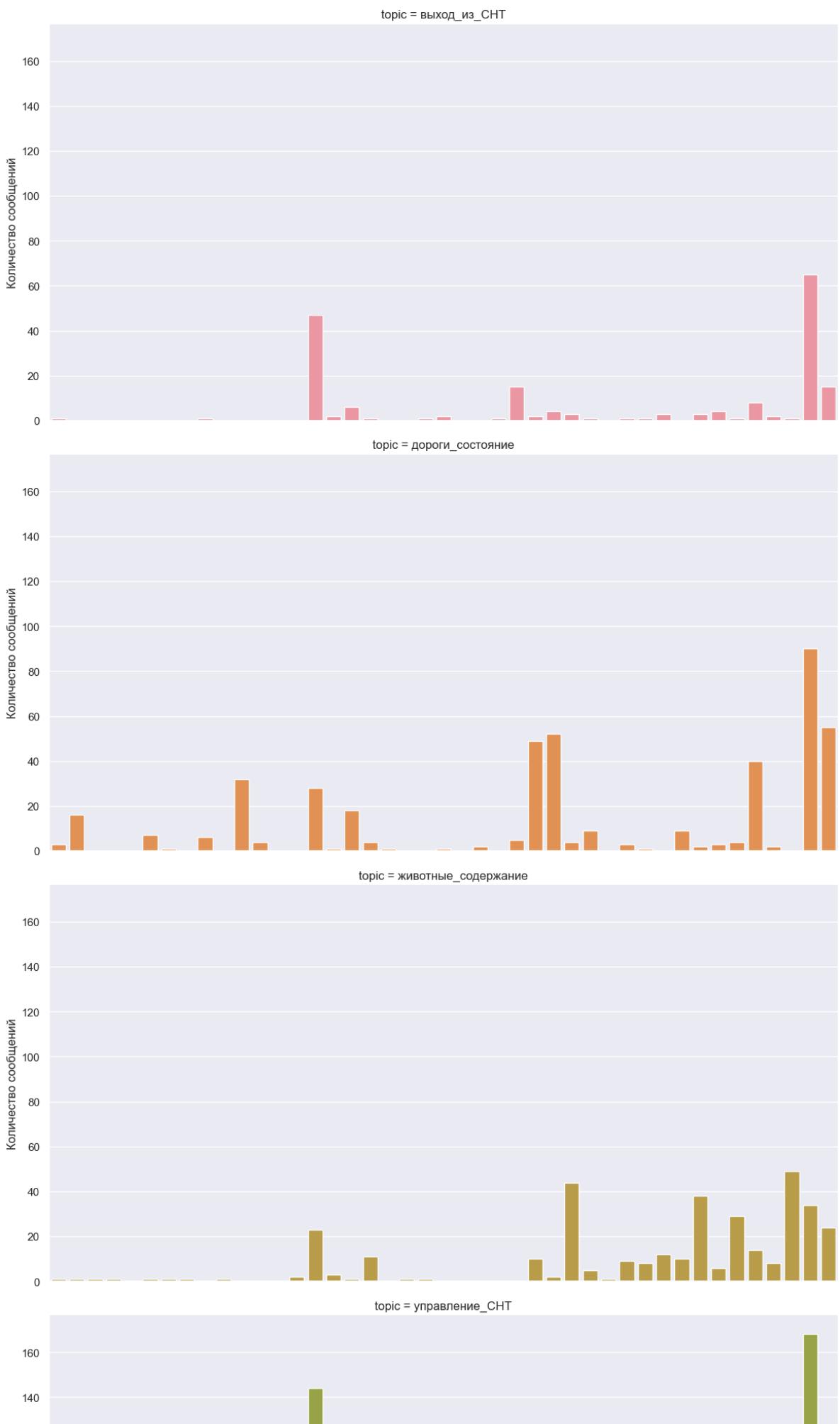
```
In [72]: # Создаём сетку столбчатых диаграмм
g = sns.catplot(data=data44,
                 x='year_month',
                 y='msg_id',
                 hue='topic',
                 col='topic',
                 col_wrap=1,
                 height=6,
                 aspect=2,
                 kind='bar',
                 sharex=True,
                 dodge=False
)

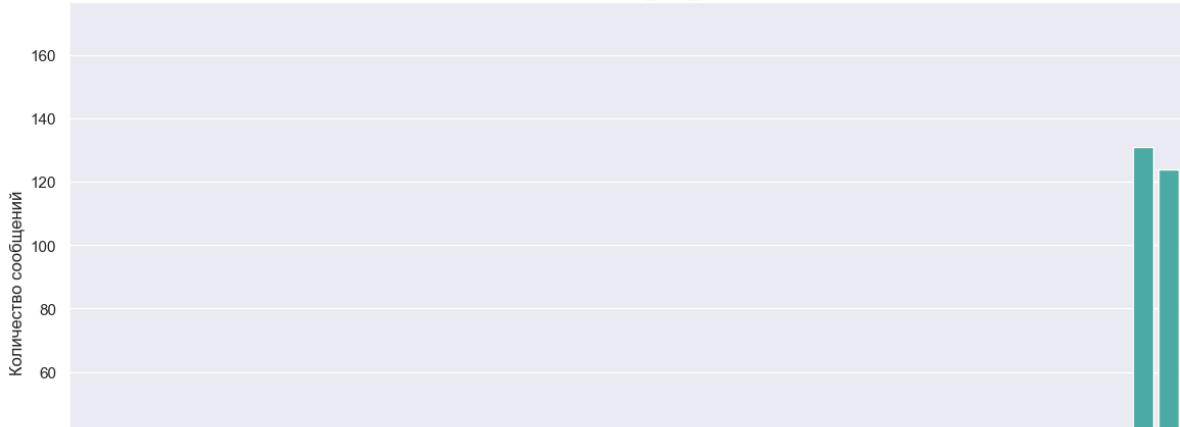
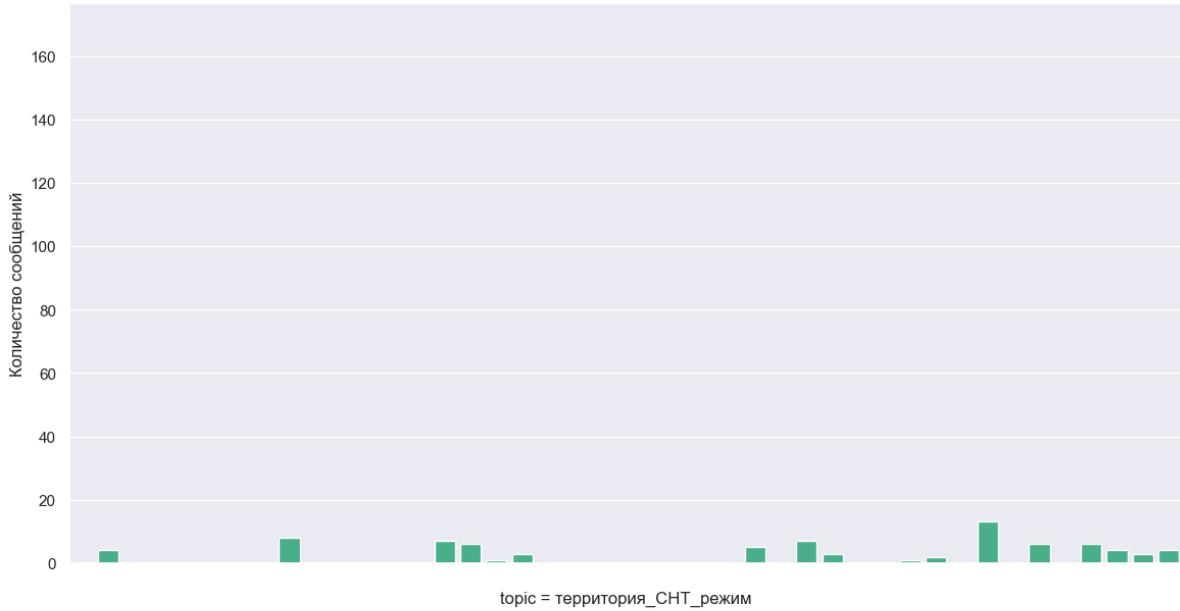
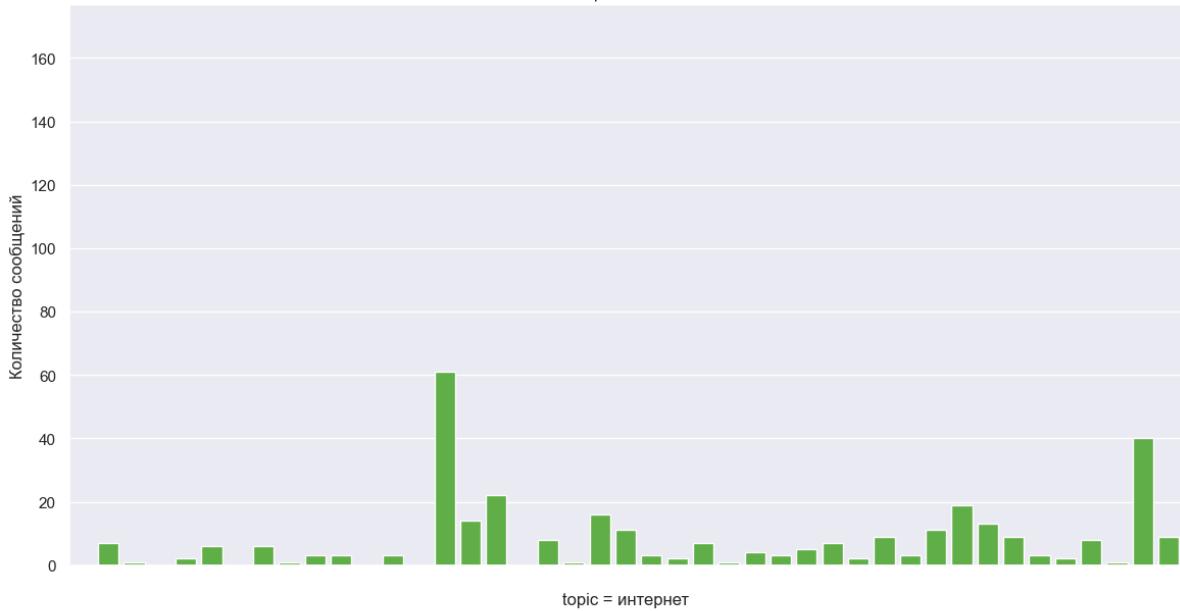
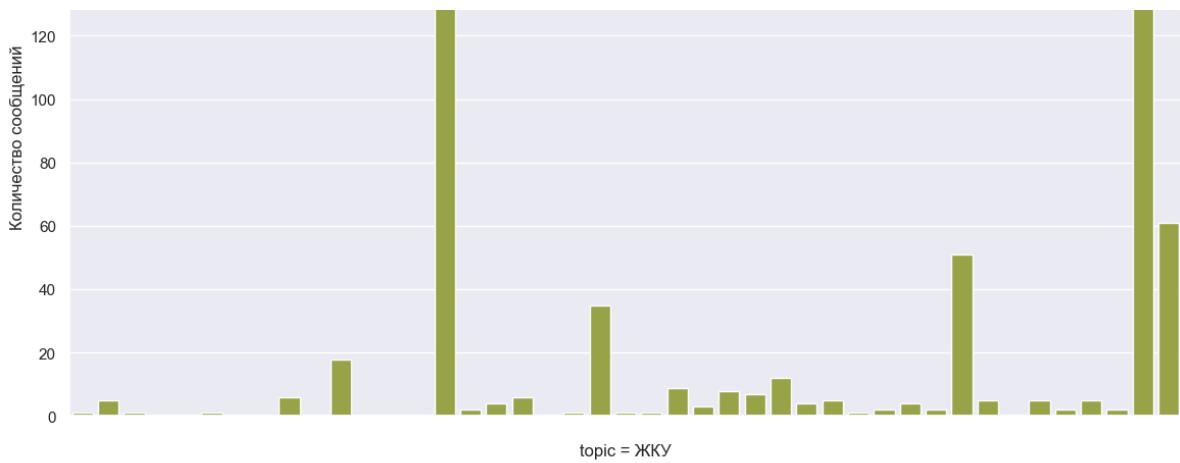
# g.fig.subplots_adjust(hspace=0.4)

dummy = plt.xticks(rotation=90)
dummy = g.set_xlabels('Год и месяц', fontsize=12)
dummy = g.set_ylabels('Количество сообщений', fontsize=12)
dummy = plt.suptitle("Динамика обсуждаемых тем", y=1.01, fontsize=14)

plt.show()
```

Динамика обсуждаемых тем





5. Определение доверительного интервала для заинтересованности членов СНТ в актуальных темах с уровнем доверия 0,95 (95%)

5.1. Расчёт погрешности

Расчёты ниже исходят из предположения, что уровень активности участников в чате, отражает уровень активности членов СНТ в целом. Определим погрешность для таких расчётов, исходя из объема выборки в 221 человек (участники чата) из всех членов СНТ.

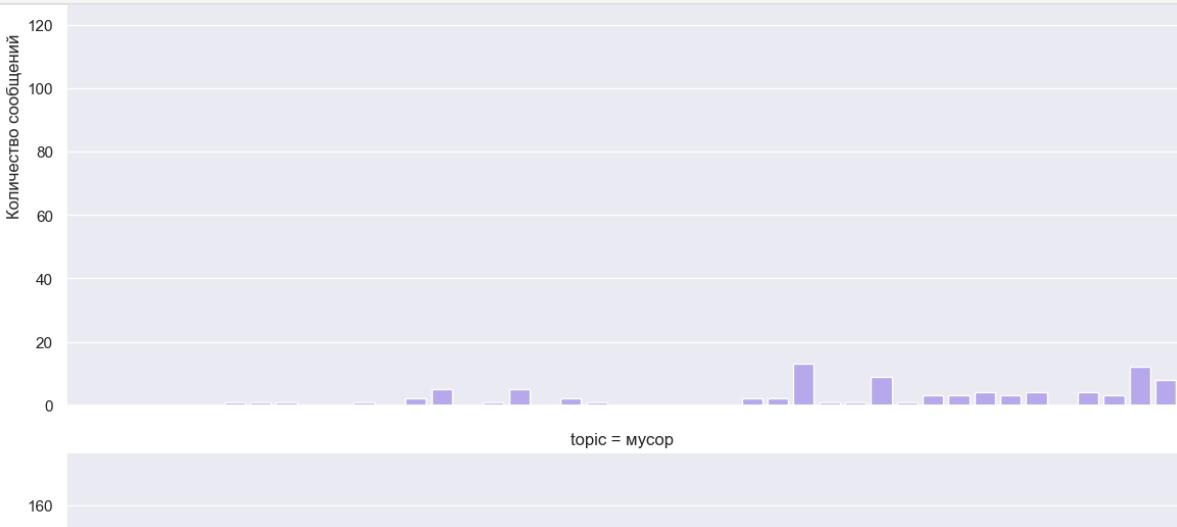
Формула для расчёта размера выборки: $n = \frac{z^2 p(1-p)}{\text{error}^2}$

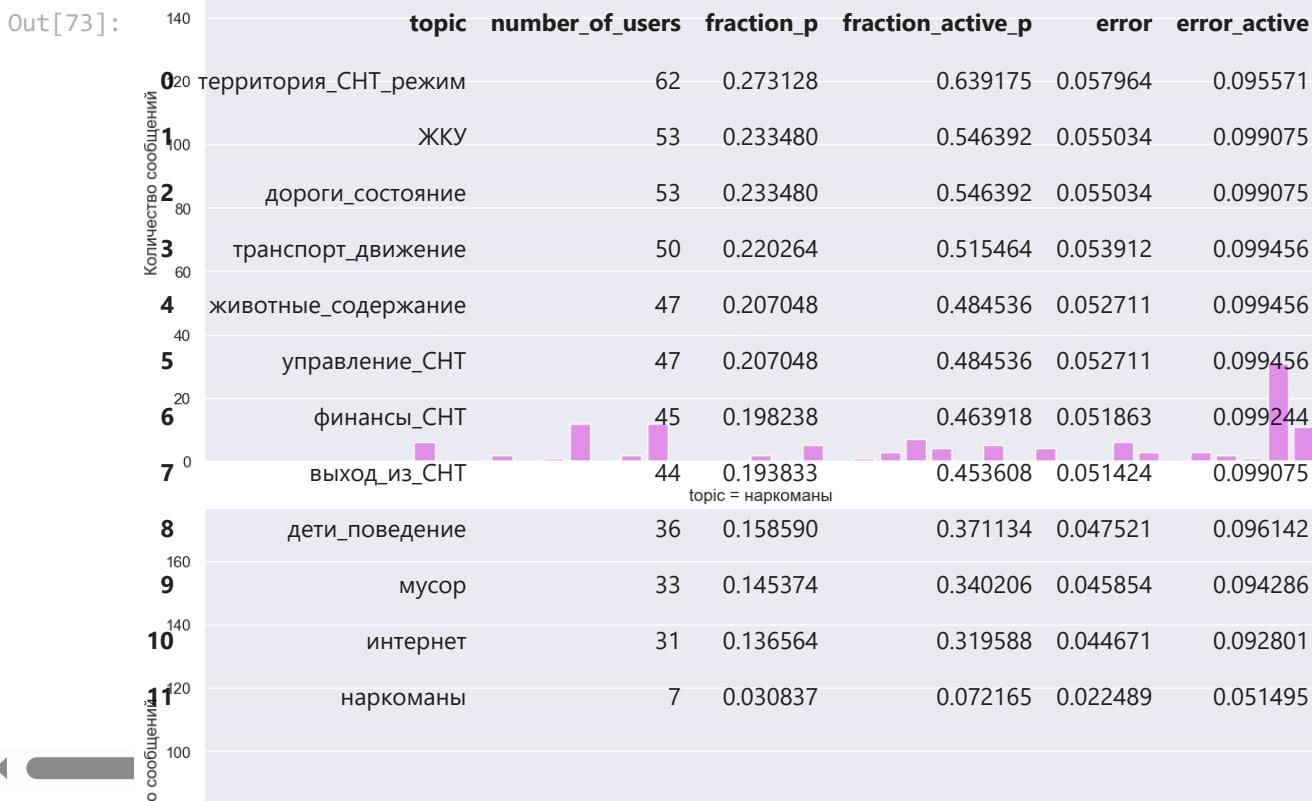
Отсюда погрешность будет равна: $\text{error} = \sqrt{\frac{z^2 p(1-p)}{n}}$

Для уровня доверия в 95%, стандартная нормальная величина $z = 1.96$

```
In [73]: # расчёт уровня погрешности для выборки в n_users из числа членов СНТ с уровнем доверия 0.95
# Берём за основу df44 с количеством участников, заинтересованных в каждой теме.
```

```
df51 = (
    df44
    .assign(fraction_p = df44.number_of_users / n_users,
            fraction_active_p = df44.number_of_users / n_active_users,)
)
df51 = (
    df51
    .assign(error = np.sqrt((1.96 ** 2 * df51.fraction_p * (1 - df51.fraction_p)) /
                           error_active = np.sqrt((1.96 ** 2 * df51.fraction_active_p * (1 - df51.fraction_active_p)) /
                           / n_active_users)
    )
    .reset_index(drop=True)
)
df51
```





In [74]:

```
# расчёт уровня погрешности для выборки доли активных участников из числа членов СНТ
err_active_frac = \
    np.sqrt((1.96 ** 2 * (n_active_users / n_users) * ((n_users - n_active_users) / n_users)) / n_active_users)

err_active_frac
```

Out[74]:

0.06435389370719749

5.2. Доверительные интервалы для заинтересованности всех членов СНТ

In [75]:

```
print(f"Доли участников чата, обсуждавших актуальные темы в разрезе тем из всех тем")

# Берем за основу df51 с количеством участников по каждой теме, их долей от числа участников и размером погрешности

# Добавим столбец с долями незаинтересованных участников
df52 = (
    df51
    .assign(fraction_q = (n_users - df51.fraction_p) / n_users)
)

# Подсчитаем стандартные ошибки
df52 = (
    df52
    .assign(standard_error = np.sqrt(df52.fraction_p * df52.fraction_q / n_users))
)

# подсчитаем доверительный интервал
df52 = (
    df52
    .assign(fraction_diapason_95 = df52.apply(
        lambda x: st.norm.interval(0.95, x.fraction_p, x.standard_error), axis=1)
)
```

```

df52_res = (
df52[
    ['topic', 'fraction_diapason_95', 'error']
]
    .reset_index(drop=True)
)
df52_res

```

Доли участников чата, обсуждавших актуальные темы в разрезе тем из всех 227 участников чата

Out[75]:

	topic	fraction_diapason_95	error
0	территория_СНТ_режим	(0.20518288699974085, 0.3410726196081887)	0.057964
1	ЖКУ	(0.17065456715081978, 0.29630578527208773)	0.055034
2	дороги_состояние	(0.17065456715081978, 0.29630578527208773)	0.055034
3	транспорт_движение	(0.15924091317438044, 0.281287721186853)	0.053912
4	животные_содержание	(0.1478823489510053, 0.26621456734855414)	0.052711
5	управление_СНТ	(0.1478823489510053, 0.26621456734855414)	0.052711
6	финансы_СНТ	(0.1403431899278645, 0.2561325809972456)	0.051863
7	выход_из_СНТ	(0.136584236020261, 0.25108096221762444)	0.051424
8	дети_поведение	(0.10680319256845378, 0.21037742417163435)	0.047521
9	мусор	(0.09579061651572784, 0.19495828216268624)	0.045854
10	интернет	(0.08850513392042952, 0.18462261938353525)	0.044671
11	наркоманы	(0.00799460330291972, 0.05367940550765297)	0.022489

In [76]:

```

# Выведем итоги по занинтересованности всех членов СНТ в актуальных темах
print("ВЫВОД")
for i in range(0, len(df52_res), 1):
    print(f"С погрешностью в {df52_res.error[i]:.1%} можно утверждать:\n"
          f"от {df52_res.fraction_diapason_95[i][0]:.1%} до {df52_res.fraction_diapason_95[i][1]:.1%}\n"
          f"готовы обсуждать тему '{df52_res.topic[i]}'.\n"
    )

```

ВЫВОД

С погрешностью в 5.8% можно утверждать:
от 20.5% до 34.1% членов СНТ готовы обсуждать тему 'территория_СНТ_режим'.

С погрешностью в 5.5% можно утверждать:
от 17.1% до 29.6% членов СНТ готовы обсуждать тему 'ЖКУ'.

С погрешностью в 5.5% можно утверждать:
от 17.1% до 29.6% членов СНТ готовы обсуждать тему 'дороги_состояние'.

С погрешностью в 5.4% можно утверждать:
от 15.9% до 28.1% членов СНТ готовы обсуждать тему 'транспорт_движение'.

С погрешностью в 5.3% можно утверждать:
от 14.8% до 26.6% членов СНТ готовы обсуждать тему 'животные_содержание'.

С погрешностью в 5.3% можно утверждать:
от 14.8% до 26.6% членов СНТ готовы обсуждать тему 'управление_СНТ'.

С погрешностью в 5.2% можно утверждать:
от 14.0% до 25.6% членов СНТ готовы обсуждать тему 'финансы_СНТ'.

С погрешностью в 5.1% можно утверждать:
от 13.7% до 25.1% членов СНТ готовы обсуждать тему 'выход_из_СНТ'.

С погрешностью в 4.8% можно утверждать:
от 10.7% до 21.0% членов СНТ готовы обсуждать тему 'дети_поведение'.

С погрешностью в 4.6% можно утверждать:
от 9.6% до 19.5% членов СНТ готовы обсуждать тему 'мусор'.

С погрешностью в 4.5% можно утверждать:
от 8.9% до 18.5% членов СНТ готовы обсуждать тему 'интернет'.

С погрешностью в 2.2% можно утверждать:
от 0.8% до 5.4% членов СНТ готовы обсуждать тему 'наркоманы'.

5.3. Доверительный интервал для доли активных членов СНТ (готовых принимать участие в обсуждении актуальных тем)

In [77]:

```
# Подсчитаем для активных участников позитивную долю, негативную долю,
# стандартную ошибку и доверительный интервал
p_frac = n_active_users / n_users
n_frac = 1 - p_frac
se = math.sqrt(p_frac * n_frac / n_users)
diapason_95 = st.norm.interval(0.95, p_frac, se)

print(f"С погрешностью в {err_acive_frac:.1%} можно утверждать:\n"
      f"от {diapason_95[0]:.1%} до {diapason_95[1]:.1%} членов СНТ "
      f"являются активными (готовы высказываться по текущим вопросам СНТ более
```

С погрешностью в 6.4% можно утверждать:
от 36.3% до 49.2% членов СНТ являются активными (готовы высказываться по текущим в опросам СНТ более 2x раз)

5.4. Доверительные интервалы для заинтересованности активных членов СНТ

```
In [78]: print(f"Доверительные интервалы заинтересованности активных членов СНТ в актуальных темах, расчитанные на основе {n_active_users} активных участников чата")  
  
# Берем за основу df51 с количеством участников по каждой теме, их долей от числа и размером погрешности  
  
# Добавим столбец с долями незаинтересованных участников  
df54 = (  
    df51[  
        ['topic', 'number_of_users', 'fraction_active_p', 'error_active']  
    ]  
    .assign(fraction_active_q = (n_active_users - df51.fraction_active_p) / n_active_users)  
)  
  
# Подсчитаем стандартные ошибки  
df54 = (  
    df54  
    .assign(standard_error = np.sqrt(df54.fraction_active_p * df54.fraction_active_q))  
)  
  
# подсчитаем доверительный интервал  
df54 = (  
    df54  
    .assign(fraction_act_diapason_95 = df54.apply(  
        lambda x: st.norm.interval(0.95, x.fraction_active_p, x.standard_error), axis=1)  
)  
)  
  
df54_res = (  
    df54[  
        ['topic', 'fraction_act_diapason_95', 'error_active']  
    ]  
    .reset_index(drop=True)  
)  
df54_res
```

Доверительные интервалы заинтересованности активных членов СНТ в актуальных темах, расчитанные на основе 97 активных участников чата

Out[78]:

	topic	fraction_act_diapason_95	error_active
0	территория_СНТ_режим	(0.48059957547611376, 0.7977509399878037)	0.095571
1	ЖКУ	(0.3997060910250033, 0.6930774141296357)	0.099075
2	дороги_состояние	(0.3997060910250033, 0.6930774141296357)	0.099075
3	транспорт_движение	(0.37296736799349023, 0.6579604670580561)	0.099456
4	животные_содержание	(0.34635841098953074, 0.6227137539589229)	0.099456
5	управление_СНТ	(0.34635841098953074, 0.6227137539589229)	0.099456
6	финансы_СНТ	(0.328697323302414, 0.5991377282439778)	0.099244
7	выход_из_СНТ	(0.3198917932742873, 0.5873247015710735)	0.099075
8	дети_поведение	(0.25013128709859844, 0.492136754138515)	0.096142
9	мусор	(0.22433634970289362, 0.456076021431127)	0.094286
10	интернет	(0.20727189901102955, 0.43190335872092916)	0.092801
11	наркоманы	(0.018725278663392318, 0.12560461824382416)	0.051495

In [79]:

```
# Выведем итоги по занимательности активных членов СНТ в актуальных темах
print("ВЫВОД")
for i in range(0, len(df54_res), 1):
    print(f"С погрешностью в {df54_res.error_active[i]:.1%} можно утверждать:\n"
          f"от {df54_res.fraction_act_diapason_95[i][0]:.1%} до "
          f"{df54_res.fraction_act_diapason_95[i][1]:.1%} активных членов СНТ "
          f"готовы обсуждать тему '{df54_res.topic[i]}'.\n")
```

ВЫВОД

С погрешностью в 9.6% можно утверждать:
от 48.1% до 79.8% активных членов СНТ готовы обсуждать тему 'территория_СНТ_режим'.

С погрешностью в 9.9% можно утверждать:
от 40.0% до 69.3% активных членов СНТ готовы обсуждать тему 'ЖКУ'.

С погрешностью в 9.9% можно утверждать:
от 40.0% до 69.3% активных членов СНТ готовы обсуждать тему 'дороги_состояние'.

С погрешностью в 9.9% можно утверждать:
от 37.3% до 65.8% активных членов СНТ готовы обсуждать тему 'транспорт_движение'.

С погрешностью в 9.9% можно утверждать:
от 34.6% до 62.3% активных членов СНТ готовы обсуждать тему 'животные_содержание'.

С погрешностью в 9.9% можно утверждать:
от 34.6% до 62.3% активных членов СНТ готовы обсуждать тему 'управление_СНТ'.

С погрешностью в 9.9% можно утверждать:
от 32.9% до 59.9% активных членов СНТ готовы обсуждать тему 'финансы_СНТ'.

С погрешностью в 9.9% можно утверждать:
от 32.0% до 58.7% активных членов СНТ готовы обсуждать тему 'выход_из_СНТ'.

С погрешностью в 9.6% можно утверждать:
от 25.0% до 49.2% активных членов СНТ готовы обсуждать тему 'дети_поведение'.

С погрешностью в 9.4% можно утверждать:
от 22.4% до 45.6% активных членов СНТ готовы обсуждать тему 'мусор'.

С погрешностью в 9.3% можно утверждать:
от 20.7% до 43.2% активных членов СНТ готовы обсуждать тему 'интернет'.

С погрешностью в 5.1% можно утверждать:
от 1.9% до 12.6% активных членов СНТ готовы обсуждать тему 'наркоманы'.

6. ВЫВОДЫ

Анализ неофициального чата СНТ в WhatsApp производился за период 10/09/2019 10:53:00 по 04/14/2023 18:48:00 по сохранённой копии чата в формате *.txt.

Сводная статистика

1. Общее количество сообщений: 6131
2. Число пользователей: 227
3. Число активных участников (отправивших в чат более 2-х сообщений): 97
4. 10% наиболее активных участников составляет 23 участника, от каждого из них отправлено свыше 86 сообщений.
5. Среднее количество сообщений от одного участника равно 27. При этом более 50% участников не написали ни одного сообщения. 40% участников написали, более 4-х. Максимальное количество сообщений от одного участника составляет 599.
6. В чате сохранены сообщения за 1312 дней, что составляет почти 187,4 недель, или 3,6 года.

7. В среднем в месяц в чат отправляется 139 сообщений. Большую часть времени в ежемесячное число сообщений не превышало 100. Максимальное число сообщений было отправлено в чат в декабре 2020 года (более 750 – в то время многие участники чата высказывали своё недовольство работой правления и председателя, а также очень частыми отключениями электричества) и в марте 2023 года (более 650).
8. В среднем в месяц в чат пишет 26 участников. В период до декабря 2020 года, а также в период с апреля по ноябрь 2021 года, число участников, оставлявших сообщения в чате в течение месяца, не превышало средней величины. Обсуждение работы правления и частные отключения электричества в декабре 2020 года вызвало увеличение активности участников чата. В том месяце их число составило почти 45. Отмечаются «пики» в количестве участников, оставляющих сообщения в течение месяца. Они приходятся на декабрь 2020 года (почти 45 участников), февраль 2022 года (около 40 участников), сентябрь 2022 года (более 45), декабрь 2022 года (45) и март 2023 года (около 85).
9. В среднем один и тот же участник отправляет в чат 4 сообщения в месяц. Большую часть истории чата количество сообщений от одного и того же участника в месяц колебалось на уровне 4. В декабре 2020 года от одного и того же участника пришлось почти 16 сообщений и почти 10 за август 2022 и неполный апрель 2023 года.
10. Относительно большая часть сообщений отправлена в чат в четверг и в пятницу. В понедельник и вторник, а также в воскресенье, большая часть сообщений отправлена в первой половине дня, со среды по субботу – во вторую. В четверг и пятницу выделяются часы активности в обеденное время и в период с 17 до 21 часа, в пятницу также в 13 часов.
11. Количество участников, пишущих сообщения в чат по дням недели распределено относительно равномерно. Больше всего участников пишут сообщения в чат со вторника по пятницу, меньше всего – в понедельник, субботу и воскресенье.
12. Наибольшее среднее количество сообщений от одного и того же участника в чате приходится на 10 часов утра вторника, вечерние часы четверга, 18 часов пятницы, 10 утра воскресенья.

Анализ содержания сообщений

1. Общий словарь значимых слов, использованный в сообщениях, составляет 7419 слов. Это достаточно много для такого чата.
2. Подавляющее число значимых слов (более 7070) встречается публикациях менее 20 раз.
3. 340 слов используется менее 20 раз.
4. 50% слов используется не более 1 раза
5. Есть слова, используемые очень часто: от 300 до 860 раз.
6. В 50% сообщений слова встречаются в среднем не более 77 раз
7. Более 600 сообщений содержит вложенные медиафайлы.
8. Общее количество позитивно воспринимаемых слов в словарном запасе чата составляет 233 слова, то есть 3.14%
9. Общее количество негативно воспринимаемых слов в словарном запасе чата составляет 45 слова, то есть 0.61%

10. До 80 позитивно-окрашенных слов встречается в чате крайне редко, единичные позитивные слова встречаются от 25 до 230 раз.
11. 17 негативных слов встречается в чате не более 1 раза, до 5 раз встречается от 4 до 9 слов. Единичные негативные слова встречаются от 6 до 20 раз.
12. Общее количество сообщений, содержащих позитивные слова 1734, то есть 27.5%
13. Общее количество сообщений, содержащих негативные слова 110, то есть 1.7%

Анализ актуальных тем

Для такого анализа определён перечень актуальных тем: 'выход_из_СНТ', 'дети_поведение', 'дороги_состояние', 'животные_содержание', 'ЖКУ', 'интернет', 'мусор', 'наркоманы', 'территория_СНТ_режим', 'транспорт_движение', 'управление_СНТ', 'финансы'.

Для каждой темы определён набор ключевых слов в формате «тема: [список ключевых слов].»

- 'выход_из_СНТ': ['город', 'ижс', 'луч', 'москва', 'посёлок'],
- 'дети_поведение': ['бегать', 'велосипед', 'гонять', 'гулять', 'дети', 'квадроцикл', 'ребенок', 'самокат', 'скейтер'],
- 'дороги_состояние': ['асфальт', 'гололед', 'гравий', 'грязь', 'дорога', 'засыпать', 'канава', 'ливневка', 'ливнёвка', 'обочина', 'подсыпать', 'покрытие', 'посыпать', 'пыль', 'разъезд', 'снег', 'сугроб', 'щебенка', 'щебень', 'щебёнка', 'яма'],
- 'животные_содержание': ['бездомный', 'выгул', 'животное', 'кот', 'кошка', 'кусать', 'кусаться', 'намордник', 'напасть', 'непривитый', 'ничейный', 'ошейник', 'поводок', 'прививать', 'прививка', 'привитый', 'собака', 'стерилизация', 'стерилизовать', 'укус', 'укусить'],
- 'ЖКУ': ['вода', 'водопровод', 'газ', 'газопровод', 'свет', 'освещение', 'обслуживание', 'электричество', 'электроэнергия'],
- 'интернет': ['интернет', 'передача', 'провайдер', 'соединение'],
- 'мусор': ['банка', 'вонь', 'вонять', 'выбрасывать', 'выбросить', 'вывоз', 'выкидывать', 'выкинуть', 'вынести', 'выносить', 'крыса', 'мусор', 'пакет', 'пластиковый', 'помойка', 'раздельный', 'стекло'],
- 'наркоманы': ['закладка', 'наркоман', 'наркотик'],
- 'территория_СНТ_режим': ['видеокамера', 'видеонаблюдение', 'ворота', 'въезд', 'забор', 'замок', 'калитка', 'камера', 'наблюдение', 'охрана', 'охранник', 'проход', 'проходить', 'сторож', 'территория', 'шлагбаум'],
- 'транспорт_движение': ['автомобиль', 'большегрузный', 'грузовой', 'машина', 'парковать', 'парковка', 'скоростной', 'скорость', 'быстро', 'не стись', 'пронестились', 'пронестись'],
- 'управление_СНТ': ['повестка', 'правление', 'председатель', 'протокол', 'решение', 'решить', 'собрание', 'устав'],
- 'финансы_СНТ': ['аудит', 'аудиторский', 'бухгалтерия', 'бухгалтерский', 'бюджет', 'взнос', 'долг', 'задолженность', 'зарплата', 'оплата', 'плата', 'платить', 'расход', 'смета', 'ставка', 'сумма', 'тариф', 'цена']

1. Среди ключевых слов более 100 раз употребляются слова «дорога», «собрание», «собака», «правление», «председатель», «свет».

2. По количеству упомянутых ключевых слов актуальные темы ранжируются следующим образом:

- управление_СНТ: 582,
- территория_СНТ_режим: 486,
- дороги_состояние: 452,
- животные_содержание: 353,
- ЖКУ: 326,
- финансы_СНТ: 239,
- выход_из_СНТ: 186,
- транспорт_движение: 191
- мусор: 161,
- дети_поведение: 93,
- интернет: 83,
- наркоманы: 10.

1. В целом участники чата упоминают в своих сообщениях не более 50 ключевых слов. У отдельных участников (6 человек) количество упомянутых ключевых слов, определяющих актуальные темы составляет от 111 до 628.

2. По доли участников, обсуждавших актуальные темы в общем количестве участников чата от 20 до 28% приходится на темы:

- территория_СНТ_режим – 27,3%,
- ЖКУ – 23,3%,
- дороги_состояние – 23,3%,
- транспорт_движение – 22,0%,
- животные_содержание – 20,7
- управление_СНТ – 20,7%,

1. По доле участников, обсуждавших актуальные темы в общем количестве активных участников чата (то есть отправивших более 2 сообщений в чат), указанные темы набирают:

- территория_СНТ_режим – 63,9%,
- ЖКУ – 54,6%,
- дороги_состояние – 54,6%,
- транспорт_движение – 51,5%,
- животные_содержание – 48,5
- управление_СНТ – 48,5%,

Определение доверительного интервала для заинтересованности членов СНТ в актуальных темах с уровнем доверия 0,95 (95%)

Используя статистические методы анализа выборки, на основе данных чата определены параметры заинтересованности всех членов СНТ в обсуждении указанных актуальных тем. Значение степени уверенности было взято на уровне 95%.

1. При заданном общем числе участников чата, погрешность определения долей членов СНТ, заинтересованных в обсуждении актуальных тем составляет от 2,2 до 5,8% в зависимости от темы.

С учетом погрешности и с учётом ранжирования актуальных тем, как указанно выше, число членов СНТ готовых обсуждать актуальные темы колеблется по разным темам в диапазонах: от 0.8% - 5.5% (тема 'наркоманы) до 20.5% - 34.1% (тема 'территория_СНТ_режим')

Это означает, что в случае проведения общего собрания членов СНТ со 100%-участием ни одна из указанных «топовых» тем, не будет обсуждена и половиной участников СНТ, и вряд ли по какой из них сможет быть принято решение.

1. С погрешностью в 6.4% можно утверждать, что от 36.3% до 49,2% членов СНТ являются активными (готовы высказываться по текущим вопросам СНТ более 2x раз).

Можно предположить, что очно в общих собраниях членов СНТ принимают участие только активные члены СНТ.

1. Для активных членов СНТ было выявлено: при заданном числе активных участников чата погрешность определения долей активных членов СНТ, заинтересованных в обсуждении актуальных тем составляет от 5,1 до 9,6% в зависимости от темы.

С учетом погрешности и с учётом ранжирования актуальных тем, как указанно выше, число активных членов СНТ готовых обсуждать актуальные темы колеблется по разным темам в диапазонах: от 1.9% - 12.6% (тема 'наркоманы) до 48.1% - 79.8% (тема 'территория_СНТ_режим')

Это означает, что при условии, что на очное собрание СНТ приходят только активные члены СНТ, большинство актуальных тем (кроме «дети_поведение», «интернет», «мусор», «наркоманы») имеют высокие шансы на полноценное обсуждение на таком собрании. Люди, очно участвующие в собрании, могут подумать, что по этим темам будут по итогам приняты какие-то решения. Тем не менее, как показано выше, с учётом заочно поданных голосов, принятие решений по таким темам будет весьма проблематичным (и оставляет окно для манипуляций).

Иными словами: *Если мы проведём собрание, как обычно проводим (то есть, те, кто приходят, что-то обсуждают и за что-то пытаются голосовать, а остальные по закону должны заочно проголосовать по повестке дня), то вероятность обсуждения и принятия какого-то решения по актуальным темам маловероятна, так как ни по одной теме мы не переваливаем за 50% участников (в доверительном диапазоне 95% и с указанной погрешностью, безусловно, есть вероятность, что что-то обсудим, но она очень мала). В указанном уровне доверия и с указанной погрешностью, максимум, что мы имеем - это 34,1% по теме территории СНТ.*

Мы исходим из того, что неактивная часть членов СНТ на собрания обычно не ходит. Если вдруг на собрание придут ВСЕ активные члены СНТ (то есть те, кто готов хотя бы пару реплик высказать по актуальным темам), то на таком собрании скорее всего будут обсуждены часть тем, как минимум те, где нижняя граница доверительного интервала ближе к 50%, а верхняя перекрывает его с большим запасом - это не более первых 8 тем из списка. Причем, заметим, погрешность для выборки из наших данных находится на уровне почти 10%. Даже если такое собрание

очно за что-то проголосует, то решение всё равно будет зависеть от заочных голосов тех, кто не пришёл на собрание. А им, как мы видели выше - ни одна тема не интересна. Именно здесь начинаются нюансы, вернее открывается большое окно для злоупотреблений. У нас же голосование не поимённое, мы никогда не сможем проверить, кто и как на самом деле голосовал. Протокол собрания подписывается председателем собрания и секретарём. Поимённые списки голосовавших не прикладываются. Поэтому, даже если активное меньшинство "продавит" на собрании обсуждение тех или иных тем, то решение по ним будет принято такое, которое выгодно нынешнему руководству СНТ.